

TABLE OF CONTENTS

	Page
INTRODUCTION	1
CHAPTER 1 DEMAND FORECAST OVERVIEW	3
1.1 The purposes of electric demand forecasting.....	3
1.1.1 Electric power characteristics of consumers.....	5
1.1.2 Demand forecast for operation power system	5
1.1.3 Demand forecast for planning power system.....	7
1.1.4 Using demand forecast for other purposes demand-side management, tariff setting, integrated renewable energy power system, and power market	7
1.1.5 Top-down and Bottom-up approach	8
1.2 Purposes of building electric demand forecast	10
1.3 Advanced metering system	14
1.3.1 Smart meter	17
1.3.2 Communications Networks and Systems.....	18
1.3.3 Integration with Information and Management Systems.....	19
1.3.4 Time-Based Rates and Demand-Side Programs	20
1.4 Demand-side management and Demand response program	23
1.4.1 Demand-side management (DSM)	23
1.4.2 Demand Response (DR)	26
1.5 Peak Demand Shaving Program	28
1.6 Renewable resources integrate into the distribution grid and Smart Grid System	30
CHAPTER 2 LITERATURE REVIEW	39
2.1 Methodologies of electrical demand forecasting	39
2.1.1 Methodology applying for electricity demand forecasting short term.....	39
2.1.2 Linear Regression Models	40
2.1.3 Artificial neural network models	41
2.1.4 Deep learning mode	42
2.2 Input data.....	43
2.2.1 Weather Variables.....	43
2.2.2 Calendar Variables.....	44
2.2.3 Historical Demand Data.....	44
2.3 Neural network structure and optimal algorithms neural network.....	46
2.4 Combining electricity demand forecast models.....	47
CHAPTER 3 MACHINE LEARNING	49
3.1 Machine learning.....	49
3.1.1 The Task, T	49
3.1.2 The Performance Measure, P	50
3.1.3 The Experience, E.....	50

3.2	Machine learning algorithms groups.....	50
3.2.1	Supervised Learning	51
3.2.2	Unsupervised Learning	51
3.2.3	Semi-Supervised Learning.....	52
3.2.4	Reinforcement Learning	53
3.3	Capacity, Overfitting, and Underfitting	53
3.4	Methods to avoid overfitting.....	55
3.4.1	Validation and Cross-validation	55
3.4.2	Early Stopping	57
3.4.3	Regularization.....	57
3.4.4	Dropout	58
3.5	Model parameters and Loss function	59
3.5.1	Model parameters and Loss function.....	59
3.5.2	A simple machine learning formula.....	61
3.6	Feature engineering.....	63
3.6.1	Rescaling.....	63
3.6.2	Standardization	64
3.6.3	Scaling to unit length	64
3.7	Linear Regression	64
CHAPTER 4 NEURAL NETWORK		71
4.1	The Perceptron Learning Algorithm	71
4.2	Activation function	73
4.2.1	Log-sigmoid activation function.....	74
4.2.2	Tanh activation function	75
4.2.3	ReLU activation function.....	77
4.3	Multilayer neural network.....	78
4.4	Gradient base Method	83
4.4.1	Gradient descent variants.....	86
4.5	First-order and second-order optimization algorithms.....	90
4.5.1	First-order optimization algorithms	90
4.5.2	Second-order optimization algorithms.....	95
4.6	Backpropagation	98
4.6.1	Concept backpropagation.....	99
4.6.2	Apply backpropagation in neural network.....	100
CHAPTER 5 APPLY DEMAND FORECAST FOR ETS BUILDING USING NEURAL NETWORK		107
5.1	Data pre-processing.....	108
5.1.1	Extract feature data	108
5.1.2	Detection and Handling of data missing and duplicate data.....	109
5.1.3	Clean up outlier.....	109
5.1.4	Adjustment Daylight saving time data.....	110
5.2	Methodological approach.....	115
5.2.1	Performance measures	115

5.2.2	Neural network model.....	116
5.2.3	Backpropagation neural network based on Levenberg Marquardt algorithm.....	118
5.2.4	Backpropagation neural network based on Bayesian regularization	118
5.2.5	Deep learning neural network.....	119
5.3	Forecasting base on Individual building	121
5.4	Two hours ahead load demand forecast.....	122
5.5	Combine forecast model	122
CHAPTER 6	RESULT AND DISCUSSION	125
6.1	Effects of input data and training technique	125
6.2	Effects of number of neurons.....	129
6.3	Single versus aggregated buildings prediction.....	130
6.4	Very short-term load forecast and short-term load forecast.....	132
6.5	Recommendations for future work	135
6.5.1	The study takes the weight of data over time	136
6.5.2	Additional data can be added to help improve the model.....	136
6.5.3	Improve algorithms and neural network model for load forecast.....	137
CONCLUSION	139
APPENDIX I	MATLAB CODE.....	139
APPENDIX II	PYTHON CODE	141
LIST OF BIBLIOGRAPHICAL REFERENCES	147

LIST OF TABLES

	Page
Table 1.1	Examples of AMI Communications Technologies19
Table 1.2	Business case summary in nominal and present value22
Table 1.3	Comparison of Dynamic rate offerings with the base rate29
Table 4.1	Three approach of the variants first-order gradient optimizers91
Table 4.2	Several second-order optimization methods in MATLAB98
Table 5.1	Daylight saving time from 2013-2019.....110
Table 5.2	Four neural network models and their hyperparameters116
Table 5.3	The result for 24 hours ahead with four neural network models in a typical week.....121
Table 6.1	Performance of learning algorithm for 7 inputs126
Table 6.2	Performance of learning algorithm for 11 inputs127
Table 6.3	Performance of learning algorithm with different neurons in the hidden layer130
Table 6.4	Performance of prediction individual building and total.....131

LIST OF FIGURES

	Page
Figure 1.1	Example bottom-up approach demand forecast9
Figure 1.2	Aggregation level for relations determining, e.g. energy demand10
Figure 1.3	Electricity energy use in Canada 201611
Figure 1.4	Breakdown of a household’s electricity use12
Figure 1.5	Residential energy use and Commercial and institutional building energy use in Canada in Canada, 2014.....13
Figure 1.6	Breakdown of \$7.9 Billion SGIG Investment15
Figure 1.7	AMI and Customer System Work Together to Automate Functions16
Figure 1.8	Three types of meter for Hydro-Quebec customer18
Figure 1.9	Meter Reading Intervals used20
Figure 1.10	The relationship between DR and EE in a DSM portfolio24
Figure 1.11	Six types of load shape apply for standard DSM load management25
Figure 1.12	Demand Response27
Figure 1.13	Generating Capacity Additions and Retirements by 2040, Reference Case31
Figure 1.14	Capacity Mix by Primary Fuel, 2015 and 2040, Reference Case.....32
Figure 1.15	Non-Hydro Renewable Capacity, Reference Case.....32
Figure 1.16	Changing Cost of Photovoltaic Solar Power Systems.....34
Figure 1.17	Montreal in Global Solar Atlas.....35
Figure 1.18	Projected Annual Energy Use with and without EE and PV Saving in New England Power Market36
Figure 1.19	Solar energy effect to daily load profile37

Figure 1.20 Overview building forecast and other factor effects38

Figure 2.1 Non-deep and Deep feed-forward neural network42

Figure 3.1 Underfitting and Overfitting.....54

Figure 3.2 Cross-validation with k fold56

Figure 3.3 Dropout Neural Net Model.....59

Figure 3.4 A Recipe for Machine Learning62

Figure 3.5 A linear regression problem67

Figure 4.1 Perceptron Learning Algorithm.....71

Figure 4.2 Neural Network describing the Linear Regression algorithm.....72

Figure 4.3 Hard Limit activation function73

Figure 4.4 Linear activation function.....74

Figure 4.5 Sigmoid activation function74

Figure 4.6 Tanh activation function.....75

Figure 4.7 Linear, Sigmoid, and Tanh activation function, along with their gradient.....76

Figure 4.8 ReLU Rectified Linear Unit activation function78

Figure 4.9 Perceptron Learning Algorithm (PLA) for fundamental binary problems79

Figure 4.10 XOR function solved by a neural network with a hidden layer80

Figure 4.11 “Artificial Neural Networks” (ANN) or “Multi-Layer Perceptrons” (MLP)81

Figure 4.12 Deep Convolutional Neural Networks –Alex Net.....83

Figure 4.13 A recipe for machine learning84

Figure 4.14 Gradient Descent Method.....86

Figure 4.15	Performance of three gradient descent variant	88
Figure 4.16	Develop and transform of the first-order gradient optimizers.....	92
Figure 4.17	Compare Gradient Descent with physical phenomena.....	93
Figure 4.18	Nesterov update	95
Figure 4.19	Forward-Mode differentiation	99
Figure 4.20	Reverse-Mode differentiation.....	100
Figure 4.21	Backpropagation demonstration.....	103
Figure 5.1	Pearson correlation coefficient of 11 variables with one output	108
Figure 5.2	Example of a district buildings.....	111
Figure 5.3	Electricity demand in the district building in one typical year.....	112
Figure 5.4	Percentage of peak power demand over than subscribed from 2013 to 2016	113
Figure 5.5	Seasonal variations of electricity demand in the district building in a week.....	113
Figure 5.6	Weekly load profile for the five buildings.....	114
Figure 5.7	Model neural network Levenberg-Marquardt algorithm in MATLAB.....	118
Figure 5.8	Model neural network Bayesian regularization algorithm in MATLAB	118
Figure 5.9	LSTM neural network	120
Figure 6.1	The number of epochs vs. mean absolute errors for the LM and BG trained forecasting model	127
Figure 6.2	Obtained regression for LM backpropagation algorithm	128
Figure 6.3	Obtained regression for Bayesian regularization.....	128
Figure 6.4	Day-ahead forecasts of the individual building in a typical week.....	132
Figure 6.5	Day-ahead versus hour ahead in a typical day in autumn	133

Figure 6.6	Day-ahead versus hour ahead in a typical day in spring	133
Figure 6.7	Day-ahead versus hour ahead in a typical day in summer	134
Figure 6.8	Day-ahead versus hour ahead in a typical day in winter	134
Figure 6.9	The average MAPE for day-ahead and hour-ahead.....	135

LIST OF ABBREVIATIONS

AB	Canadian province of Alberta
ADAM	Adaptive moment estimation optimizer
AGC	Automatic generation control
AI	Artificial Intelligence
AMI	Advanced Metering Infrastructure
ANN	Artificial Neural Networks
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
BC	Canadian province of British Columbia
BESS	Battery Energy Storage System
BR	Bayesian regularization
BTM PV	Behind the Meter Solar Photovoltaics
CCS	Carbon Capture and Storage
CIS	Customer Information Systems
CNN	Convolutional Neural Networks
CPP	Critical-Peak Pricing
DA	Distribution Automation
DC	Direct Current
DER	Distributed Energy Resources
DLC	Direct Load Control
DR	Demand Response

DSM	Demand Side Management
EE	Energy Efficiency
ÉTS	École de Technologie Supérieure
EV	Electric vehicle
EVs	Electric vehicles
FL	Fuzzy logic
GHG	Greenhouse gas
GIS	Geographic Information Systems
GRÉPCI	Groupe de Recherche en Électronique de Puissance et Commande Industrielle
GW	Gigawatt (power)
GWh	Gigawatt-hours (energy)
HVAC	Heating Ventilation and Air-Conditioning
IEA	International Energy Agency
IEEE	Institute of Electrical and Electronics Engineers
IESO	Independent Electricity System Operator – Electrical Power Operator in ON
ISO	Independent System Operator
KW	Kilowatt
KWh	Kilowatt Hour
LDCs	Local Distribution Companies
LM	Levenberg-Marquardt
LSTM	Long Short-Term Memory
LTLF	Long-term load forecasting
MAPE	Mean Absolute Percentage Error

MDM/R	Meter Data Management/Repository
MDMS	Meter data management system (MDMS)
MLP	Multilayer Perceptron
MSE	Mean Square Error
MTLF	Medium-term load forecasting
NEB	National Energy Board
NERC	North American Electric Reliability Corporation
NPV	Net Present Value
OMS	Outage Management Systems
DSM	Distribution Management Systems
ON	Canadian province of Ontario
PDM	Peak Demand Management
PHEV	Plug-in Hybrid Electric Vehicle
PLA	Perceptron Learning Algorithm
PLC	Power-line communications
QC	Province of Quebec
R&D	Research and Development
ReLU	Rectified Linear Unit
REN21	Renewable Energy Policy Network for the 21 st Century
RET	Screen Clean Energy Management Software developed by Natural Resources Canada
RF	Radio Frequency
RMS	Root Mean Square

XXVIII

RMSE	Root Means Square Error
RNN	Recurrent Neural Networks
RTO	Regional Transmission Organization
RTP	Real-Time Pricing
S2S	Sequence to Sequence
SARIMA	Seasonal Autoregressive Integrated Moving Average
SCADA	Supervisory Control and Data Acquisition
SGD	Stochastic gradient descent
SGF	Smart Grid Fund (Ontario)
SGIG	Smart Grid Investment Grant
SME	Smart Metering Entity
STLF	Short-term load forecasting
SVM	Support Vector Machine
T	Temperature
THI	Temperature-Humidity Index
ToU	Time of Use
U.S.	United States of America
VVC	Voltage and VAR (reactive power) Control
WCI	Wind Chill Index

LIST OF SYMBOLS

$\$/kWh$ Price per kilowatt hour of electricity

% Percentage

$\text{¢}/kWh$ Cents per kilowatt hour of electricity

$^{\circ}C$ Degrees Celsius

$^{\circ}F$ Degrees Fahrenheit

Ah Ampere hour

Ca\$ Express the amount in Canadian Dollars currency

kW Kilowatt

kWh Kilowatt hour

TWh Terawatt hour

US\$ Express the amount in U.S. Dollars currency

W Watt

INTRODUCTION

Accurate forecasting of the electricity demand of buildings will help optimize the operation of the electricity distribution system. Besides, good results in the demand forecast will create opportunities for electricity users to apply electricity demand management programs and load control or load reduction at peak time. Doing this work will bring benefit both power users and power companies.

Load forecasting plays a vital role in energy management in smart buildings. It is expected that precise prediction of loads can bring significant economic benefits to smart buildings by enabling accurate demand response strategies for peak load reduction, reducing electricity use, and utilizing microgrids. This thesis presents a comprehensive and detailed study for the very-short-term and short-term forecasting of load in district buildings using the artificial neural network (ANN). We aim to evaluate four neural network training algorithms and discuss each model's ability to accurately forecast loads for a district energy system for hour-ahead and 24 hours-ahead of using weather, time variables, and historical loads as inputs. A detailed comparison study has been carried out considering four ANN-based training methods, namely Bayesian regularization, Levenberg-Marquardt backpropagation, and two ADAM algorithms with deep learning networks. The results reveal that ANN with Bayesian regulation backpropagation has the best overall root mean squared and mean absolute percentage error performance.

Furthermore, precisions obtained with hour-ahead are better than a day ahead. Almost all the models performed better, predicting the overall campus load than the single building load. Besides, it is observed that the efficiency of the ANN-based forecast model is dependent on many parameters, such as forecast model structure, input combination, activation functions, and training data.

The content of the thesis.

The thesis is divided into six chapters, with the main contents mentioned in which chapter as follows. Chapter 1 presents an overview of the purpose of electric load demand forecasting,

the role of load forecasting for each customer group, particularly for residential and buildings. The introduction part also shows some parts relevant to demand forecast like a smart meter, AMI, DSM, Integrate Renewable Energy Resource.

Chapter 2 Literature review was presented, in this chapter show some method using for demand forecasting. The contents related to the consideration of short-term load forecasting. The content related to the input for forecasting, research, and evaluation methods, load forecast results.

Chapter 3 and Chapter 4 will give an overview of Machine Learning and Neural Network.

Chapter 5 Apply short term and very short-term load forecast for district building using a neural network.

Chapter 6 Result, discussion, recommendation, and future work also presented in this chapter.

CHAPTER 1

DEMAND FORECAST OVERVIEW

1.1 The purposes of electric demand forecasting

Electric demand forecasting has a long history, almost begin with the operation of the electric power network. There are many purposes of using power load forecasting. For example, electricity forecasting demand using power system operation in short-term and real-time, forecasting electricity demand using for expanding power grids and power generation resources to meet the electricity demand. Besides, load forecasting is also used to set the electricity tariff for each type of customer. For example, we have different electricity prices for commercial customers, industrial customers, and resident customers. Furthermore, the demand forecast also uses for building policies in Demand Side Management programs, Demand Response, integrate renewable energy sources. Demand forecast using for calculating operational optimization for smart grid, and smart microgrid systems.

According to the electric load forecasting time frame, demand forecast can be divided into three types: a long-term forecast, medium-term forecast, and short-term forecast.

In each type of load forecast, each customer or utility members are interested and used the result for many different purposes. For example, in power system operation, the electricity system operator will forecast the load for schedule planning and dispatching generator resources to meet the demand for electricity. Additionally, the electricity wholesaler and retailer can use the load forecast to predict electricity prices, which helps optimize the electricity trading plan.

Two approaches are Top-down and Bottom-up for load forecast.

To forecast electric load, often use historical data to predict future periods. In some cases, when we have a new energy demand, because we do not have historical data, then we will use the data planning and simulation model to forecast electric load. For example, when there is a new factory or a new commercial center, the load forecast will be used based on the power usage plan to provide load forecasting.

For the load with historical data, two methods are used: statistical methods (ARIMA, Polynomial Regression ...) and use the artificial intelligence to forecast electrical load (SVM, Tree Decision, Neural Network...). The more development of information technology infrastructure, smart meter system, other measuring devices, the more data are available in the power system. Hence, applications in the operation electricity system are grown. The load forecasting models are also developed very quickly. Many studies have done and published in recent years. The power utilities are also very interested in this research area, and accurate load forecasts will help power utilities plan and operate more efficiently.

Although various industries have many inventories to store their products, in contrast to modern electricity technology, storing large amounts of electricity is still a challenge. Therefore, electricity must always be a balance between the amount of electricity generated and provided to the electric consumer immediately. In other words, power companies or grid power operators must continuously balance the demand and supply of electricity. The power demand forecast was used in all sections of the electricity industry.

Purposes of power demand forecasting are used for generation, transmission, and distribution power system planning, electric and energy system operation, financial planning, designing an electricity price rate, and demand-side management. Because of the significant role of demand forecasting in the operation of electricity companies, inaccurate load forecasting can lead to the financial trouble of a power company. Although power demand prediction is an essential contribution to the operation and planning of electrical systems, incorrect loading predictions can lead to device failures or even a blackout power system. In general, the limitations of electricity storage and the need for electricity use by society have exciting consequences for load forecasting, such as complex seasonal models, the need to be extremely accurate, and the 24/7 data collection on the grid [1].

Load forecasting plays an essential role in all segments of power system planning and operation. Load forecasting is a crucial function to operate an electricity network efficiently, reliably, and economically. Therefore, power demand forecasting becomes a hotly debated issue recently. It is also essential information for evaluating the profitability of investments in new technologies for power companies [2].

The value of energy demand management becomes more critical in up-to-date decades. Because many energy resources are decreasing, emissions are increasing, and the development of renewable energy and clean energy required developing globally. Demand forecasting plays an essential role in state-owned and private enterprises in managing the demand and supply balance. Therefore, the use of models to accurately predict energy using trends is a vital subject for the electric power system [3].

1.1.1 Electric power characteristics of consumers

The demand for electricity customers may change over time, because personal activities in the hourly cycle, each time of the day, will have electricity usage different. Similarly, working day and weekend also has different levels of electricity usage; the months of the year also have different uses depending on the weather of each season of the year.

Depending on the character of the load types, such as electrical resident loadings, industrial electrical loads, commercial and administrative electrical loads, will have different periods of high electricity usage. During daily, electricity resident load will be much affected by human activity time in a load day, usually high in the early morning to near noontime and early-evening time, at night time is the time to go to bed so low electric power load. Commercial electrical loads will change over time as commercial activities take place, which is also the time when there is large electricity consumption, the weekend load may be higher on weekdays. For administrative demand loads that will be closer to the administrative time, for example, the administrative time from 9h00 to 17h00, other times will consume less electrical energy. The industrial electrical load will depend on the technology line of customers using electricity. Several customers like aluminum smelters will use much electric power at night time when the prices are often lower than in another period.

1.1.2 Demand forecast for operation power system

The forecast power demand exactly is the essential function of power companies. To operate a safe and reliable electrical system, predicting the demand for the power system is a critical

task to balance supply and demand accurately. If the load prediction error is too large, it may cause inadequate power mobilization and may cause overloading of the grid or cause problems in the electric power system. Power load forecasting to meet operational objectives is often forecasted in three categories the medium, short term, and very short-term to meet operation planning or the real-time dispatch.

The short-term forecast requires information about the electric demand from one hour ahead, a day ahead up to a few weeks. Information obtained from the short-term demand forecasts is vital for making maintenance plans in short term plans, also for scheduling, dispatching generating units, electricity networks for economic and secure operation in power systems.

The load forecast is classified by time frame, and load forecast can be separated in long-term, medium-term, and short-term [4]. Short-term load forecasting impacts to the financial situation of electric companies and is the daily work of these systems. Short-term load forecasting (STLF) using implement some of the critical functions related to power system operation in the short term, from an hour to a week. Reliable load forecasting helps power companies optimally allocate power generation supplies, ensure supply and demand balance, and have a suitable reserve. Forecasting with high accuracy reduces the possibility of interrupting electricity supply for electric consumers, increasing the reliability of the electric power system.

Most of the decision or control functions presented in this work require knowledge of future conditions, most especially power demand.

Automatic generation control (AGC) algorithms require knowledge of the inertia, governor, and frequency response for the next period of operation. The response capabilities of each unit have to be known to provide sufficient response capability as demand changes. The unit participation factors should be based on the economic dispatch to follow demand changes optimally. Some new AGC packages do directly use demand forecast, as that knowledge of demand trends in the next few minutes can provide more optimal AGC control strategies. This case is especially true when renewable energy, wind, and solar, are included as such generation does not give inertia, governor, or frequency response. AGC algorithms, including knowledge of demand trends in the next few minutes, permits the government of feed-

forward or tracking controls. AGC algorithms can take into account the rate of generation increases limits, valve-point loading, prohibited zone operation, loss of renewable generation, etc. [2][5].

1.1.3 Demand forecast for planning power system

In addition, short-term load forecasts (STLF) apply for operational purposes; Long-term demand forecast is also the input data for the planned power system. Long-term demand projections annual energy usually ranges from two or five years to ten years or twenty years ago. They are designed for long-term capital investment studies. For example, in the planning of the electricity system, determine the capacity and type of new power plants to be built, the types of equipment to build power transmission lines. The accurate forecasting of electrical loads helps power companies reduce both the investment and maintenance costs of the electricity system. Grid development and power development plans based on accurate demand forecasts ensure investment efficiency; avoid wasting resources and electricity system with a reasonable reserve level.

Medium-term load forecast (MTLF) forecast from one month to five years or ten years or more. The average forecast concept is also used for fuel maintenance and planning, optimum utilization of water resources by hydroelectric plants for several years are based on monthly energy.

Underestimated demand forecasts can lead to under capacity, leading to reduce electricity service quality where electricity outages can happen. On the contrary, over predicting load forecasts can lead to inefficiencies investment for power companies [1].

1.1.4 Using demand forecast for other purposes demand-side management, tariff setting, integrated renewable energy power system, and power market

In the electricity market, the load forecast results will create a demand curve and contribute to creating the trend of electricity prices according to the law of supply and demand. Therefore, the demand forecast is also interested in many participants, such as investors in

the electricity sector, electricity wholesale, and retail enterprise participating in the electric power market.

Demand-side management is executed by several methods and be described in the section in this chapter. Demand management methods should be based on short-term load forecasts. The electric end-user side participates in changing load profile, reducing capacity used during peak periods of the day.

Moreover, load forecasting also can help power utilities in establishing electricity tariff structures that promote designing effective demand-side management programs [2].

No single forecasting technique can be considered helpful for every condition. Decide the load forecast technique depending on the type of forecast and the available input data. In some cases, to apply to combine more than one method forecast is better than the most individual accuracy one. Therefore, each power company needs to find the most suited technique for its purpose [5][6].

1.1.5 Top-down and Bottom-up approach

Load prediction can be built based on two approach's top-down and bottom-up. Top-down and bottom-up are different relates to input data. The top-down approach will take into account the aggregate data, influencing factors at a significant level in a country, region, or type of industry. For the bottom-up method, the detailed data in each small power consumption device will be taken care of, thereby building a load profile for each customer and extrapolating the level of electricity use of a similar customer group application. Nowadays, with the data collection tools of each electrical device, simulation software tools, the demand forecast has been much more convenient than before. The bottom-up method also helps to analysis the power consumption characteristics of companies, thereby helping to build load-profiles of each type of electricity user. This data is also useful for setting electricity tariffs as well as applying for energy-saving/demand management and load adjustment programs [7][8].

Figure 1.1 below illustrates the load research method to produce a load forecast of the paper industry. It can be seen as the bottom-up approach. This approach will help researchers understand the details of electrical consumption in the paper industry.

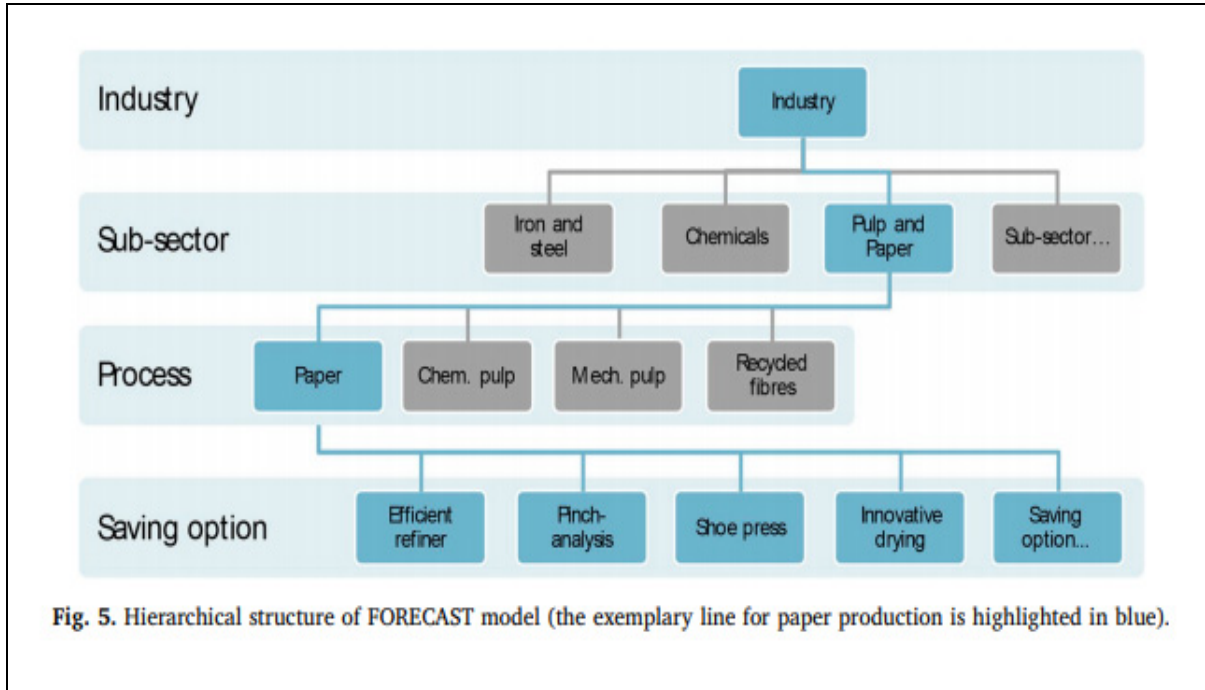


Figure 1.1 Example bottom-up approach demand forecast
Taken from T. Fleiter et al (2018)

Figure 1.2 below demonstrates two top-down and bottom-up approaches for load forecasting. Depending on the data, one of two methods can be selected to perform load forecasting or the combination of both methods.

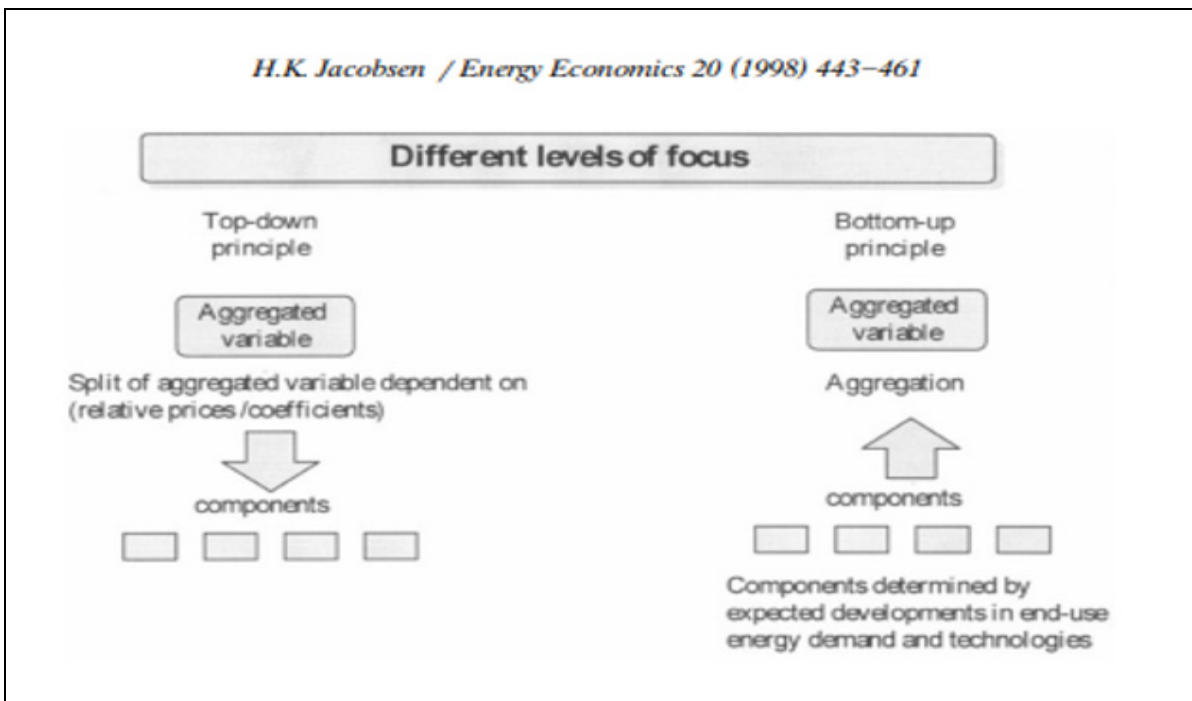


Figure 1.2 Aggregation level for relations determining, e.g. energy demand
Taken from H. K. Jacobsen et al (1998)

1.2 Purposes of building electric demand forecast

The electricity demand forecast for large buildings is important for all power utilities around the world and in Canada. The rate of this load accounts for a large proportion of the total electricity demand.

Figure 1.3 below shows the electricity used by the residential sector, accounting for nearly one-third of the electricity consumption. If appropriate policies and technology are developed, the potential for electricity saving and power use in this area is significant.

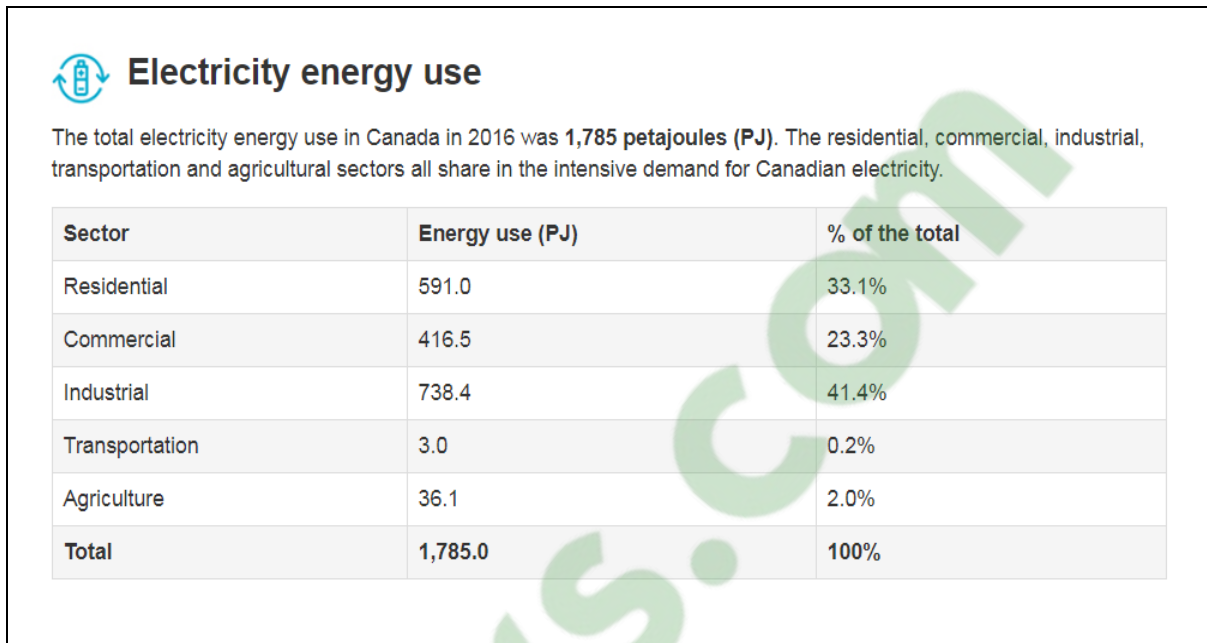


Figure 1.3 Electricity energy use in Canada 2016

Taken from Natural Resources Canada

www: <https://www.nrcan.gc.ca/science-and-data/data-and-analysis/energy-data-and-analysis/energy-facts/electricity-facts/20068#L6>
 (Retrieved from website on February 27, 2020)

The figure 1.4 below presents the percentage of electricity use in the building of Quebec, Canada. The portion of the electricity used by buildings mostly comes from hot water, heating and air-conditioning systems.



Figure 1.4 Breakdown of a household's electricity use

Taken from Hydro-Québec

www: <http://www.hydroquebec.com/residential/customer-space/electricity-use/electricity-consumption-by-use.html>

(Retrieved from website on February 27, 2020)

The same size house today uses one-third less energy than it did in 1990. Energy efficiency in household buildings improved 47 percent among 1990 and 2014, saving more than 671.6 PJ of energy and Canadians \$12.4 billion in energy expenses [9][10][11].

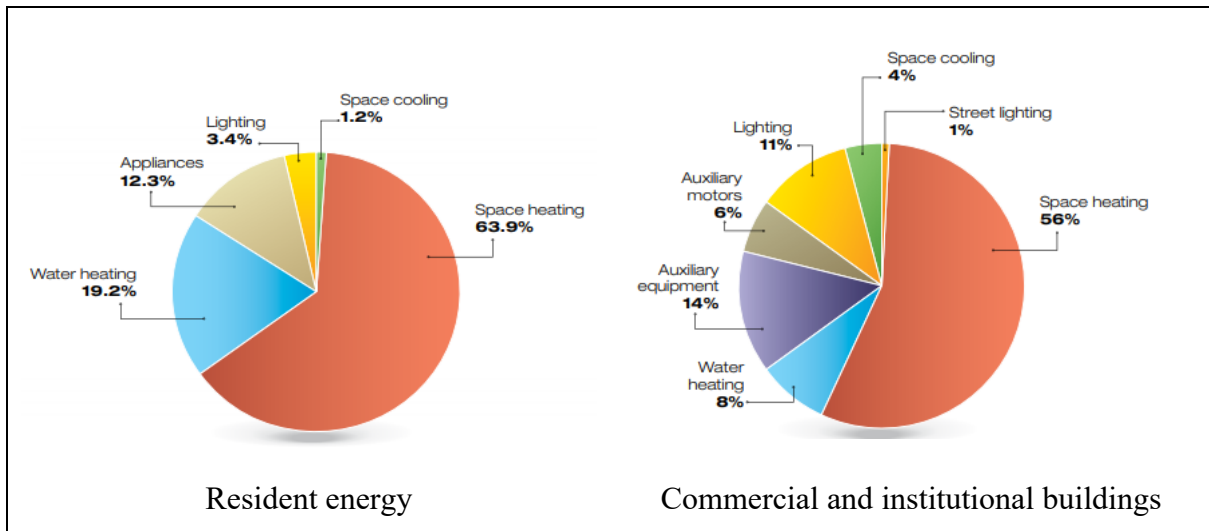


Figure 1.5 Residential energy use and Commercial and institutional building energy use in Canada in 2014
Taken from National Energy Use Database (2014)

Figure 1.5 shows that Canada has approximately 500,000 commercial and institutional buildings. In commercial and institutional buildings have 13.1 million Canadians currently work. Therefore, the potential to save electricity and optimize the use of electricity in these buildings is vast, and the human role is critical in this process [12][13].

Accurate forecasting of the electricity demand of buildings will help optimize the operation of the electricity distribution system and create opportunities for electricity users to apply electricity demand-side management programs and load control or load reduction at peak time. Doing electricity demand-side management programs will bring benefit both power users and power companies. The trend of building automation, technology to manage the information system of buildings simulation, which designing Building Information Modeling (BIM), the model smart buildings is developing rapidly. Technologies such as the Internet of Things (IoT), Big Data, 4G telecommunications network, and the upcoming 5G will support the technology development related quickly. Information technologies also increase the capacity of demand load forecasting. The demand load forecasting as well as are the input data for managing power load in the building, and this result using for optimizing the energy usage of customers [14].

The paper [15] shows that the ability of intelligent household buildings to create a collaborative network is the additional improvement and application of the idea of microgrid and intelligent buildings. In this perspective, connecting intelligent household buildings can implement a real-world opportunity for experimenting with effective building operations and helping each household green building (GB) to its neighboring GBs for sustainable energy use. However, new critical difficulties and complexities that concern energy control and reliability arise from the coordination among GBs and the main power system. Besides, the uncertainties of producing energy by renewable energy resources and the power demand forecast also make it more challenging to obtain optimal energy management [15].

1.3 Advanced metering system

Advanced measurement infrastructure (AMI) is an integrated system, and it includes smart meters, metering data management systems, and modern communication systems. Advanced measurement infrastructure permits two-way communication among power companies and electricity consumers. Many essential functions were provided by the AMI system, those functions that were previously impossible or not automated. For example, the AMI would be the capability to measure power consumption, remote connect and disconnect services, and supervise voltages.

Connected by the devices was installed on the customer side, such as indoor displays and programmable communication thermostats. AMI system additionally allows power companies to propose new time-based energy tariffs and incentives mechanics to assist consumers in reducing peak demand and managing energy consumption and electricity expenses [16].

The US Department of Energy (DOE) was started Smart Grid Investment Financing Program (SGIG) in 2009. This program funded with the US \$ 3.4 billion as an example of this significant investment from the government for modernizing the power grid infrastructure. In the report: Advanced Metering Infrastructure and Customer Systems, 2016 [16], shows substantial investment in AMI for 2009 in transmission, distribution, measurement equipment, systems customer and modern smart grid technology.

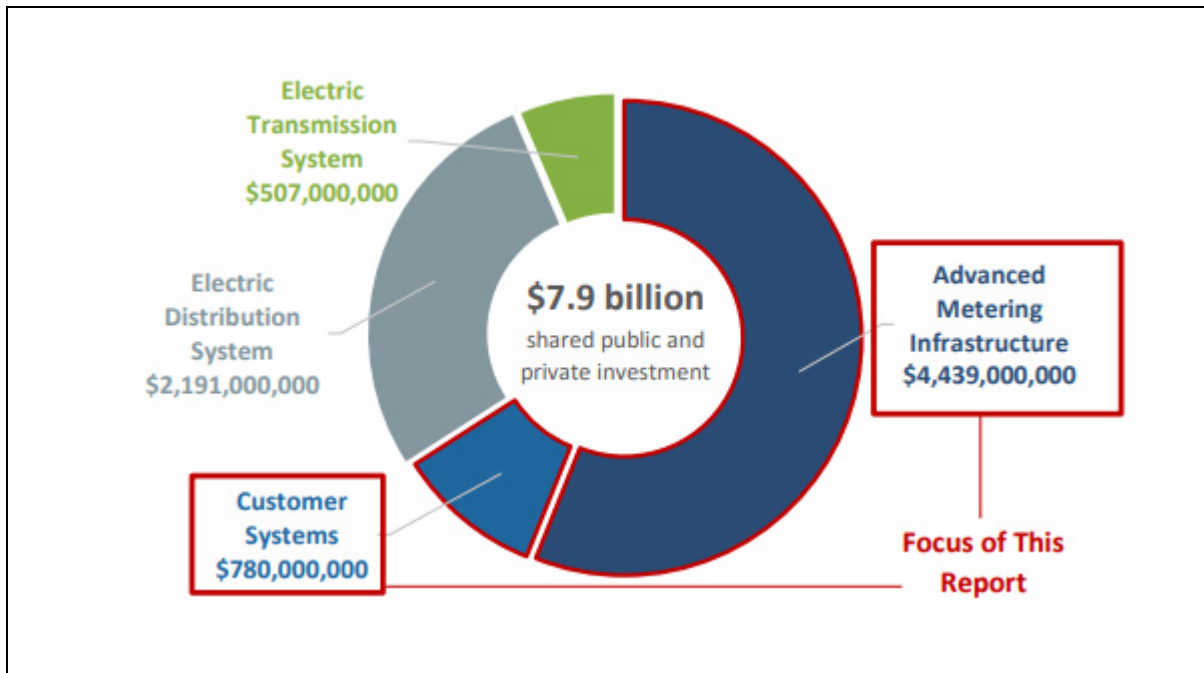


Figure 1.6 Breakdown of \$7.9 Billion SGIG Investment
Taken from the Office of Electricity Delivery and Energy Reliability
United States Department of Energy (2016)

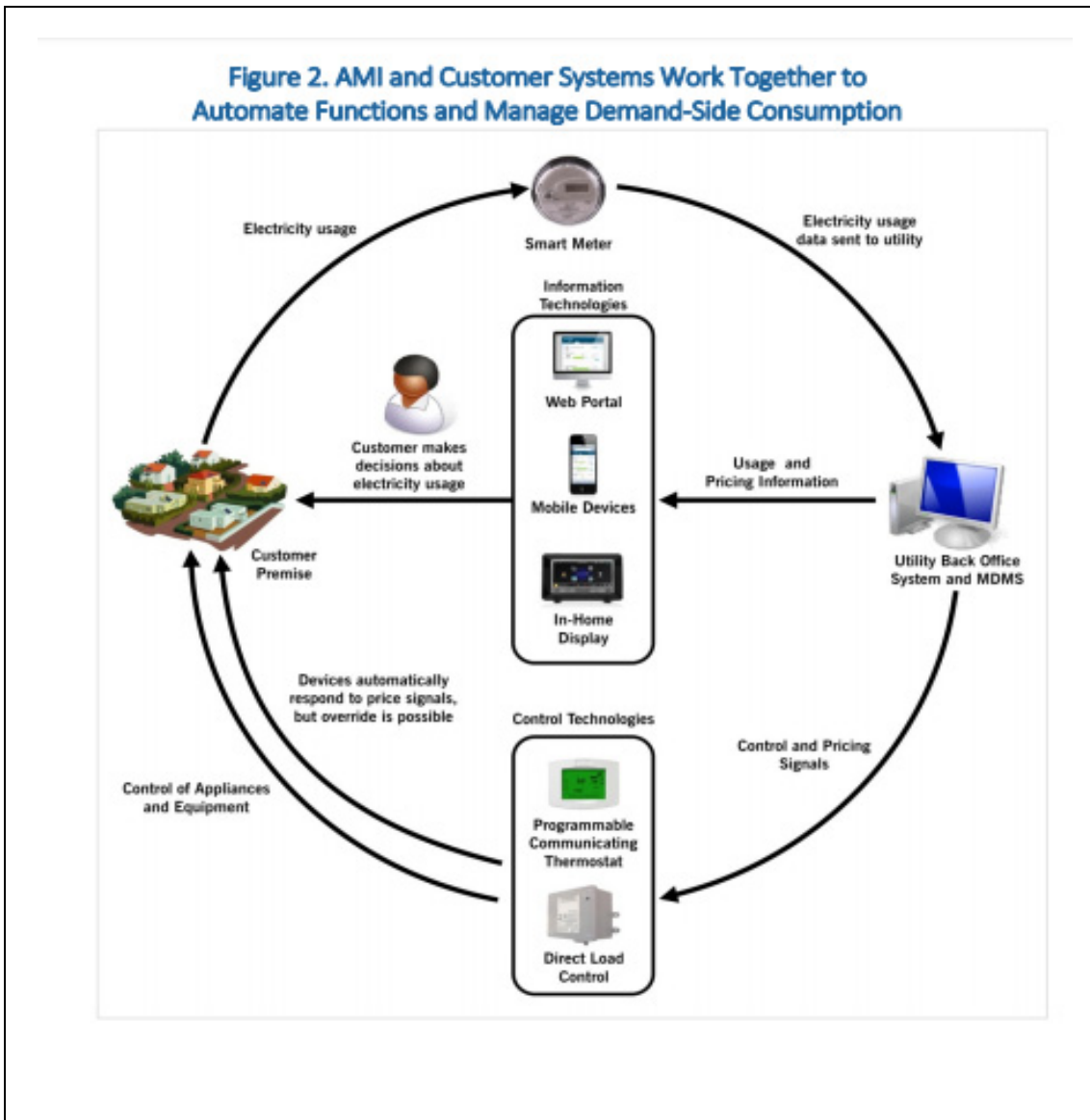


Figure 1.7 AMI and Customer System Work Together to Automate Functions and Manage Demand-Side Consumption
Taken from the Office of Electricity Delivery and Energy Reliability
United States Department of Energy (2016)

The deployment of AMI generally consists of three main components: Smart meter, communication network, and integrated with other energy management system:

The smart meter installed at the distribution grid allows power companies and customers to collect usage power data base on pre-set time, for example, every 5, 15, 30, or 60 minutes.

Vast amounts of metered data were sent via new or modernize communications systems to power utilities, which data can be used for exploitation or backup.

A metering data management system (MDMS) is the system that can collect, store, and processing of the power in the established period. This system often integrates with other essential information and control systems, including Payment Systems, Outage Management System (OMS), Geographic Information Systems (GIS), Customer Information Systems (CIS), and Distribution Management System (DMS) [16].

1.3.1 Smart meter

The smart meter is a vital device of the AMI system. The primary function of the smart meter is to measure the electricity consumption in 5, 15, 30, or 60-minute intervals of each electricity consumers. Besides, there are other functions, such as monitoring the voltage level, monitoring, control (open/close) the status to provide electricity for customers. Smart meters transfer data collected from meters to power companies for processing, analyzing, and recommending time-based electricity prices to customers. Customers can see, check the metering data, and use some functions to respond or pay for electricity bills for power companies.

For example, Hydro-Québec has installed three types of meter for electrical consumers, the meter is communicating (made by Landis+Gyr, Elster, General Electric), non-communicating (made by Itron), and an old-generation meter.

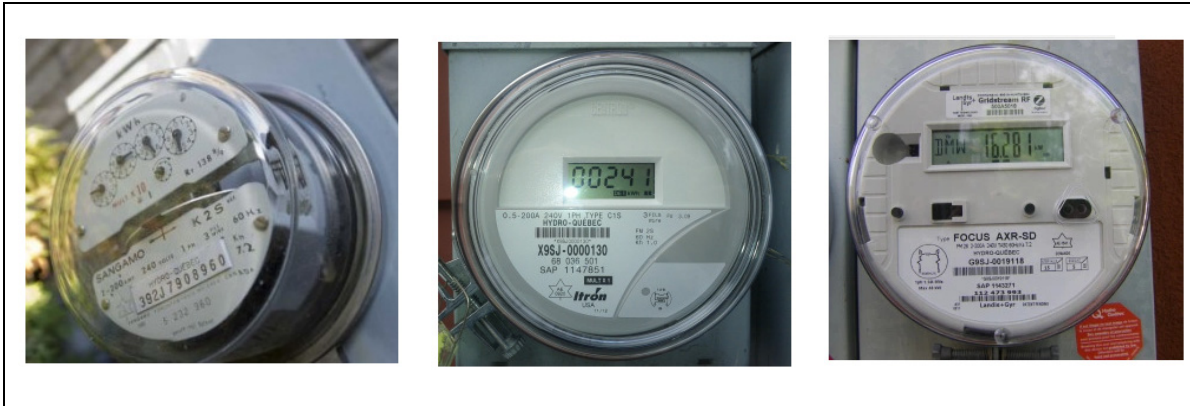


Figure 1.8 Three types of meter for Hydro-Québec customer
 Taken from <https://pointclairerefuses.wordpress.com/types-of-meters/>
 (Retrieved from website on February 27, 2020)

Hydro-Québec has modernized its meter fleet in recent years. Communication meters contribute to the primary service of Hydro-Québec and are free for all customers. The main advantage of this technology is that the meter transfers forward the electricity-use data to our systems remotely.

1.3.2 Communications Networks and Systems

The capabilities of the communication networks used in AMI can transfer large data streams accurately, reliably, and on time. These communication networks are used in addition to AMI systems to also connect to other information systems used in the distribution power system, including MDMS, CIS, OMS, and DMS.

According to the report, most of the Smart Grid Investment Allotment (SGIG) power companies are installed with modernized or upgraded communication networks to expand the AMI system. Depend on business goals, service area characteristics, and business rule constraints, power companies install and utilize a wide range of wireless and wired communications technologies. The communications technologies were shown in Table 1.1 below:

Table 1.1 Examples of AMI Communications Technologies
 Adapted from report Advanced Metering Infrastructure and Customer Systems of
 The Office of Electricity Delivery and Energy Reliability
 United States Department of Energy (2016)

Wired	Wireless
1. Fiber optic cable 2. Power line communications (PLC) 3. Telephone dial-up modem 4. Digital subscriber line (DSL)	1. Radio Frequency (RF) - the mesh network 2. RF - Point to multipoint 3. RF - Cellular

Power companies also often customized their systems, connecting numerous programs and integrations to the new network associated with many vendor products as well as concern legacy problems. To select the communication technologies and configurations best suited to the power companies needed to test multiple conditions, analyzing each smart technologies can utilize: • Bandwidth • Latency • Price • Reliability and coverage • Spectrum available • Power stored for backup • Cyber security consideration [16].

1.3.3 Integration with Information and Management Systems

Electricity companies always look for methods to extract the most valuable information from the AMI system. This data also be used in other systems of power companies. Systems connected to conventional AMI systems include the following:

- **MDMS:** Collecting, processing, storing, and managing power metering data.
- **Payment system:** Create invoices for customers based on collected measurement data.
- **CIS:** Customer information system includes information about electricity customers and electricity payment history.
- **OMS:** Processing data about the status of customers' electricity service provision.
- **GIS:** Geographic Information System to send information to groups of grid repair workers.

- DMS:** Distribution Management System to processing data when operation and maintenance system, planning power outages plan, and control voltage levels of customers to perform the process of optimizing reliability and regulating the voltage and reactive power (VAR). Therefore, integrating AMI with information systems to improve the efficiency of power companies is essential.

On-demand, the power companies are setting the interval and time to collect energy data from the meter.

For large customers, usually commercial and industrial, power companies usually set the recording time and reading time to 15 minutes. These customers usually pay electricity bills, including components energy and demand components.

For residential customers, electricity companies are most used within an hour's reading period, enough for payment purposes. Most power companies use their website when giving to consumers on power using information.

Figure 1.7 reveals the number of meter readings utilized by SGIG projects. Consumers can obtain or request invoice information on demand utilizing the website or call customer service in case of high billing or abnormal consumption profiles [16].

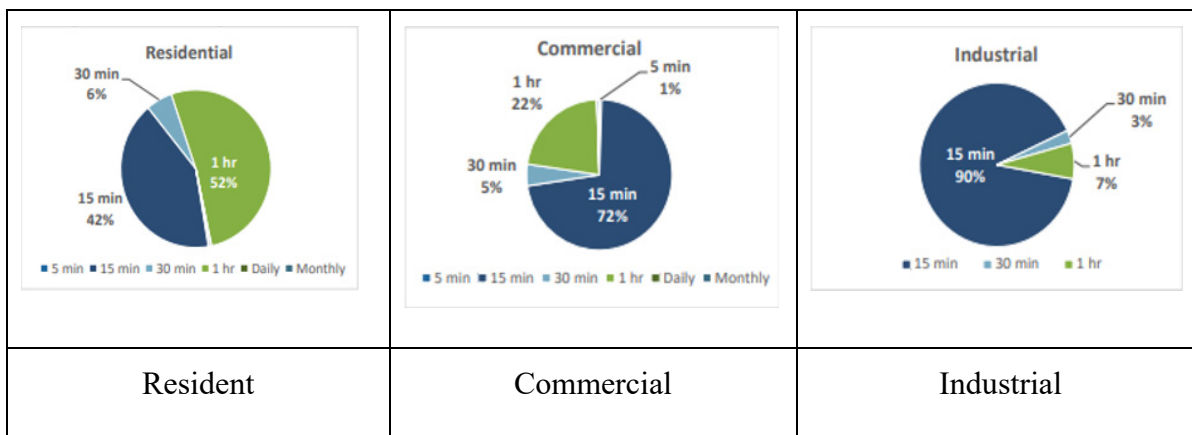


Figure 1.9 Meter Reading Intervals used
 Taken from the Office of Electricity Delivery and Energy Reliability
 United States Department of Energy (2016)

1.3.4 Time-Based Rates and Demand-Side Programs

The implementation of AMI technologies has helped companies and customers to have information about power consumption in each interval. Connecting AMI systems while, communication with other systems have provided power companies with new capabilities to reduce electricity consumption, especially during peak periods. Through two programs, voluntary incentives and a time-based electricity pricing mechanism. The two mechanics to encourage customers easy to apply and bring benefits to both sides, the customers, and the electricity company.

Time-based pricing programs are available in many different formats. They offer a range of different rates that change by the hour in the day, weekday or weekend, month, or seasons. Some electric companies offer incentive-based programs instead of time-based programs to achieve the required goals [16].

In the case of Ontario, Canada, the province was the first to install smart meters and apply usage rates, allowing small and large utility customers (> 50 kW) to manage their electricity consumption actively. Besides, the smart meter also generates large amounts of consumption-related data, which can be used to stimulate new values.

The IESO has appointed a smart metering entity (SME) in Ontario. The implementation and operation in the Meter Data Management/Repository (MDM/R) in the whole of Ontario were responded by IESO. MDM/R is a central hub providing a common platform for storing, processing, authenticating, and managing hourly electricity usage information to support the payment processes of local distribution companies with a highly secure environment.

With nearly five million smart meter sending hourly data to MDM/R and more than 60 LDCs (Local Distribution Companies) integrated into the system, the MDM/R in Ontario is one of the world's largest sharing systems, adding 100 to 120 million records per day.

British Columbia Hydro has started the Smart Metering Program since 2011. BC Hydro's Smart Metering Program has established a modern infrastructure, including replacing outdated meters from existing customers by the smart metering system. Furthermore, this

comprehensive system that helps BC Hydro manage a safe, reliable, and efficient power system. The smart meter program has been successful when it recovered its investment by reducing the amount of stolen electricity, efficiently operating the grid, and energy saving. Significant benefits of the Smart Metering Program for electric consumers will:

- Improve customer service by creating more accurate electricity bill reports, simplifying the process of opening and closing accounts of electricity users, and eliminate invoices estimate.
- Reducing power theft reduces costs - costs up to about \$ 100 million each year.
- Enhance operational effectiveness and decrease lost power by voltage optimization. More economical operating values are transferred back to all BC Hydro customers.
- Assistance consumer selection and switch by providing direct and up-to-date information about the energy their use.
- Help improve British Columbia's electrical network through reinstall old meters and building an infrastructure platform to support customer generation resources, electric vehicles, and microgrids.
- The financially significant portion of the Smart Metering Program shown in Table 1.2 is a positive NPV of \$ 520 million [17].

Table 1.2 Business case summary in nominal and present value
of Smart Metering Program in BC Hydro
Taken from BC Hydro (2011)

BUSINESS CASE SUMMARY IN NOMINAL AND PRESENT VALUE		
Business Case Summary	Nominal Value (\$M)	Present Value (\$M)
Gross Benefits attributable to Smart Metering Program, less costs related to the achievement of individual benefits	\$4,658	\$1,629
Less: Ongoing operating and maintenance expenses and incremental asset replacement capital	(745)	(330)
Less: Smart Metering Program Costs	(930)	(779)
Total Net Value for the period F2006 to F2033	\$2,983	\$520

1.4 Demand-side management and Demand response program

1.4.1 Demand-side management (DSM)

The electricity demand continues to increase, to ensure balance to the demand and supply of electricity, more generation capacity to meet the rising demand are required. Besides, economically, the marginal cost of electricity investment always increases (fuel costs and operating costs, investments...). Therefore, to achieve the optimal technical and economic situation, the responsibility be on both supply and demand sides. For power companies, investment and operation power system optimal are the most important. On the power consumption side, that is the demand side, the demand for electrical energy must be reduced by managing consumption through DSM programs.

The power companies design, plan, implement and evaluate the DSM programs to assist customers in modifying their electricity load profile for both demand (kW) and energy (kWh) in the timing and level. Figure 1.10 below shows the relation between DR and DSM [18].

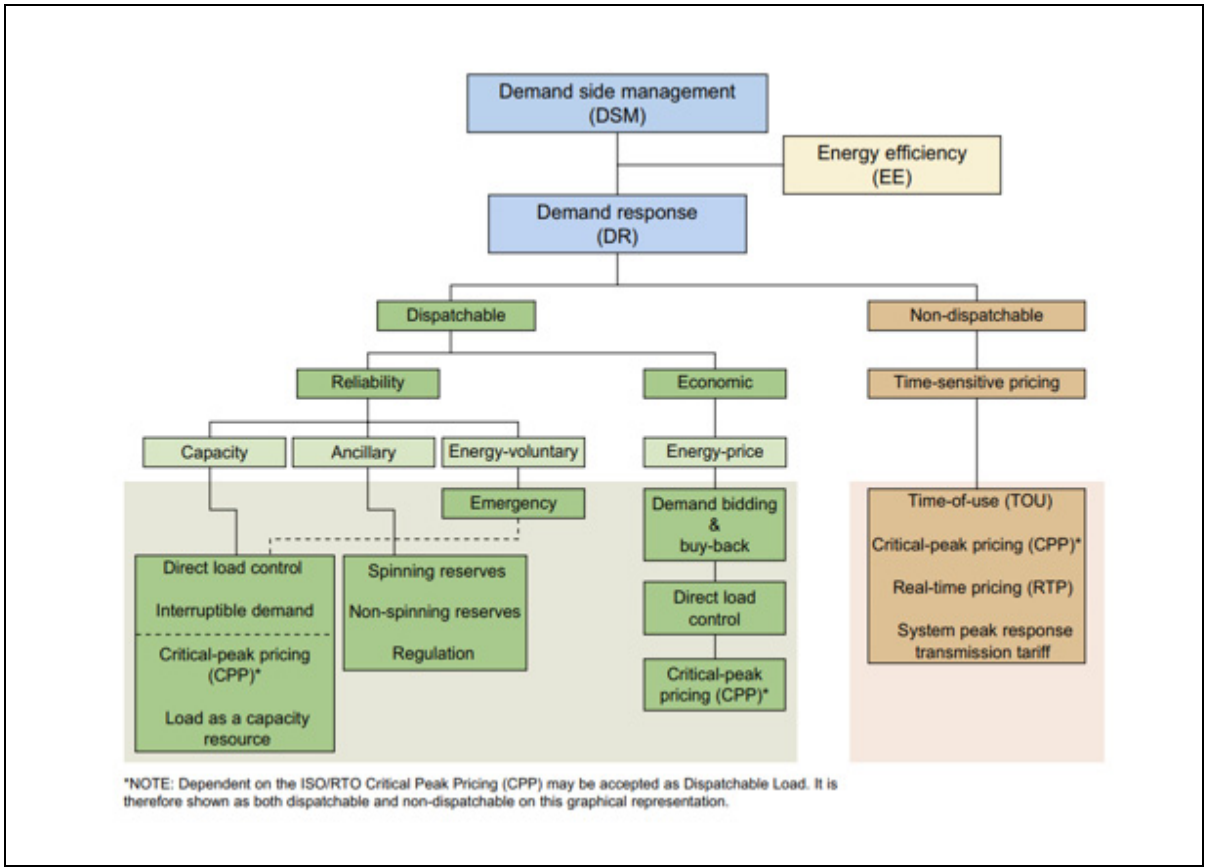


Figure 1.10 The relationship between DR and EE in a DSM portfolio
 Taken from W. Prindle et al (2012)

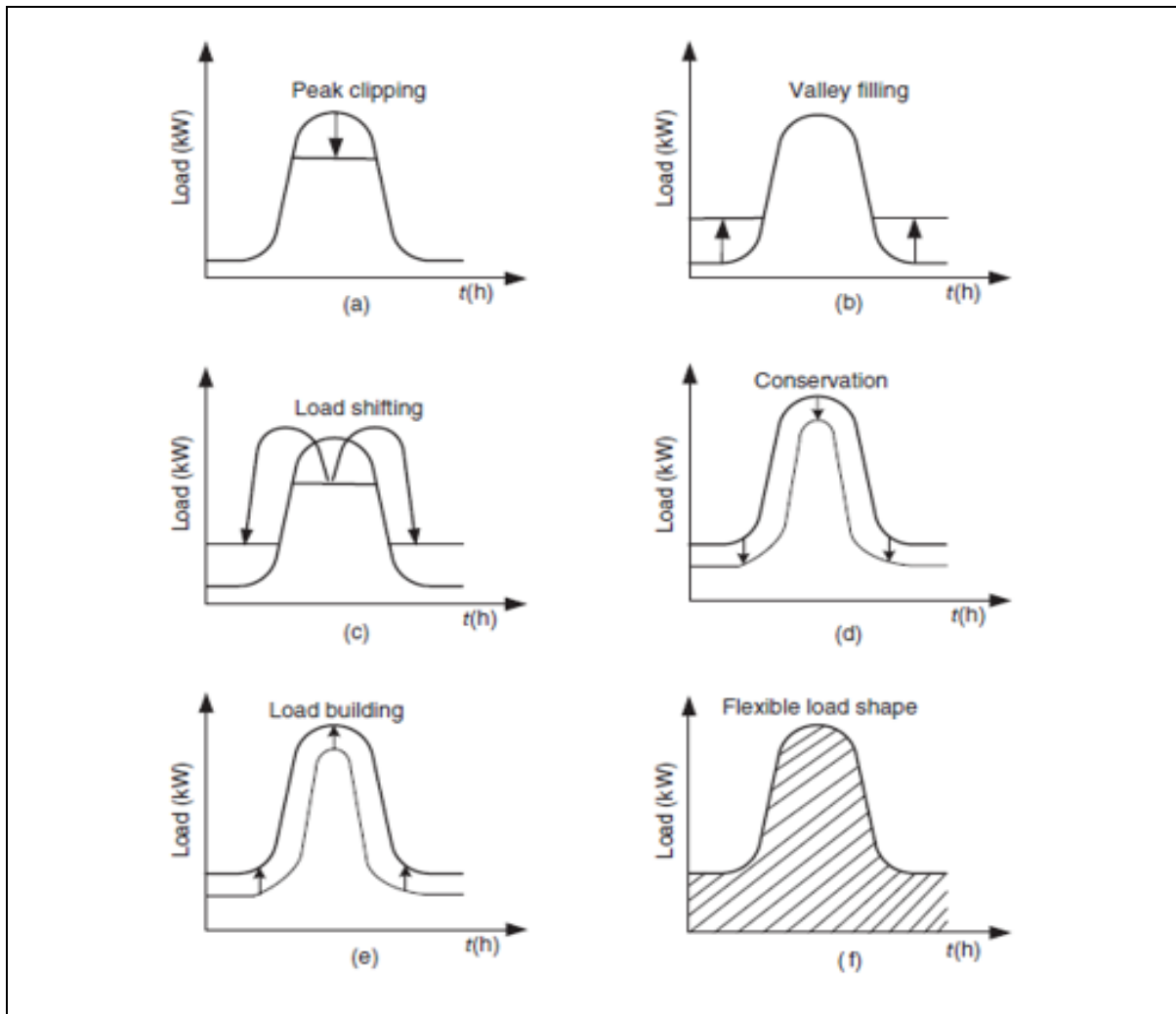


Figure 1.11 Six types of load shape apply for standard DSM load management
Taken from A. A. Sallam et al (2019, p. 435)

Figure 1.11 shows six components applies for the DSM program. We can see all relevant with the load profile of the electric consumer. In other words, the demand forecast is an important part of success in the DSM program [19].

Key Variables Affecting the Management of the Canada Electric System are shown in the report Vision 2050 [20]: the following variables affect how Canada future electricity system will be managed and designed:

- 1) Energy efficiency and demand management.
- 2) Electric vehicles (EVs).

- 3) Consumer management of energy.
- 4) Grid modernization (Smart Grid).
- 5) Human resources.

At the province level in Canada, the Quebec government is making public its energy transition policy, which puts customers at the lead of approaching actions by the year 2030.

The significant targets to be achieved by 2030:

- 1) ENHANCE by 15% energy efficiency.
- 2) REDUCE by 40% the number of petroleum products used.
- 3) ELIMINATE the use of coal for thermal generators.
- 4) INCREASE by 25% overall renewable energy resource product.
- 5) INCREASE by 50% production from bioenergy [21].

We can see that at the federal and provincial levels have clear and specific policies to promote DSM projects.

1.4.2 Demand Response (DR)

The International Energy Agency (IEA) shows that at the end-user level, deployment of demand-side compliance has been limited. Like other forms of traditional compliance, it is mostly centralized and assigned to large industrial or commercial consumers as well as several programs targeting heating services through night-time tariffs.

Total demand response program is through traditional schemes such as arrangements to interrupt service at critical times, or drastically changed day-time and night-time tariffs, is around 40 gigawatts (GW), approximately amounting to 0.5% of global electricity generation capacity using today.

The number of digital devices connected energy-related is growing exponentially. Digital devices are the main reason to increase the opportunity for customer participation in power systems. For example, devices that can participate in the DSM program include electric vehicle (EV) charging, distributed generation renewable energy sources, and energy storage

systems behind the meter. In addition, providing greater flexibility through various small-scale devices can enhance the overall capacity of the electrical system.

Several authorities around the world are intensifying their efforts to meet DSM programs, and China and Ireland regard DSM as an essential pillar to increase renewable power generation in their most modern plan.

First, countries need to develop the first smart grid infrastructure related to smart meters. Smart meter program investment reaches a record of nearly \$ 20 billion in 2017, a four-fold rise of 2010.

Demand response can be both non-dispatchable and dispatchable.

“Non-dispatchable demand response” is a program based on retail price designs to offer customers a choice. That can change over time at high prices at peak times and lower at other times. Customers will decide whether to reduce consumption during periods of high demand.

“Dispatchable demand response” meaning planned consumer shifts that consumers agree to take. It includes direct load control devices of consumers, such as appliances for heating, water heating, and air conditioning. Direct load control can reduce direct consumption in peak time and return to using in off-peak time.

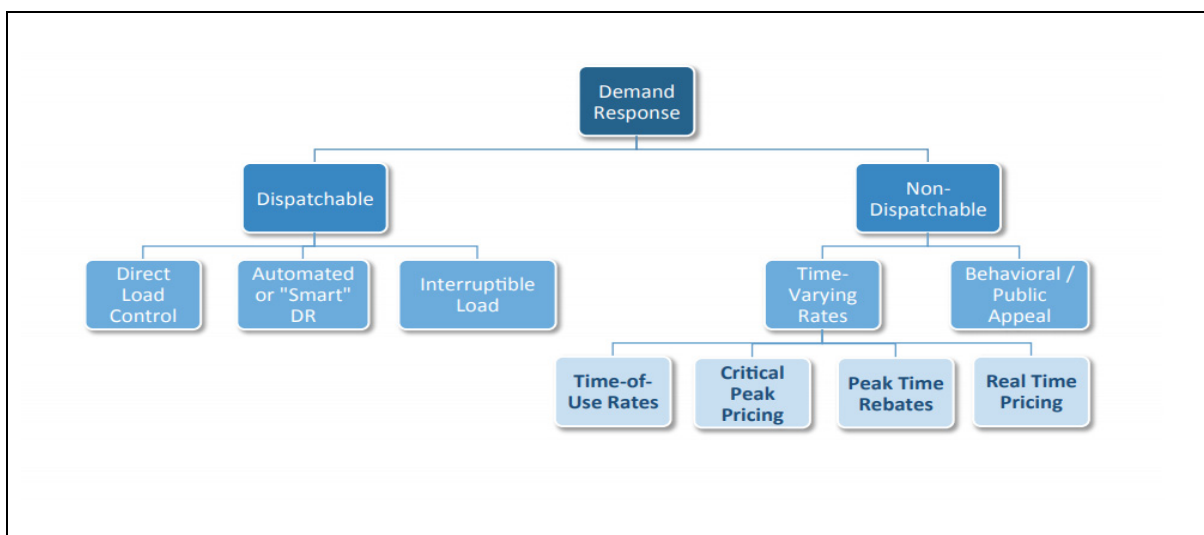


Figure 1.12 Demand Response
Taken from Asa S. Hopkins (2017)

The critical connection between the Demand Response Program and the Smart Grid can encourage home energy storage systems, such as the rooftop photovoltaic system with intelligent batteries system and electric vehicles, to shift charging times to off-peak times. Demand-responsive and smart grids are interconnected in many application areas. Many of the benefits associated with investing in smart grids and demand response, such as better management of electricity consumption, can help to operate optimally and efficiently through responding to electricity price signals or dispatch from grid operators. Besides, investing in smart grid technologies, like a smart meter, helps activate demand-responsive programming. Smart Grid can facilitate real-time information coordination from generation supplies, distributed energy resources (DER), and demand response. Smart grid and demand response will bring greater efficiency to the electrical system through exchange communication and grid integration between power companies. Finally, the smart grid will support consumer activities and allow consumers to manage energy costs better [22].

1.5 Peak Demand Shaving Program

By participating in initiatives launched under the Hydro-Québec Energy Savings Plan, customers can help Hydro-Québec overcome energy-saving targets and meet the challenge of decreasing requirement capacity. New programs and initiatives proposed financial incentives to reduce consumers' energy use during peak times without sacrificing comfort. For example, under the Residential Load Curtailment Program, customers may allow Hydro-Québec to disconnect their electric water heater for a short time several times a year. However, the water in the tank still hot for a long time; This will not influence customers' daily habits.

The goal of the program is to reduce 300MW capacity demand by 2020. Regard the energy savings plan (2003-2015) More than 25 programs and initiatives for the residential, organization, commercial, and industry customers to improve energy conserving.

- Awareness-raising, market transformation, and R & D activities.
- Savings of 8.8 TWh, the equivalent of 500,000 households' energy use, and 10% more than the initial goal of 8.0 TWh.

- \$ 1.7 billion investment, including approximately \$ 900 million in direct financial support to clients [14].

For example, McGill University has registered in Hydro-Québec's Peak Demand Management (PDM) program. During freezing climate events, Hydro-Québec will call for its customers to reduce power demand, so reducing the pressure on the province's power infrastructure. At McGill, thirteen buildings are enrolled on downtown campus: Genome, Life Sciences, McIntyre Medical, Elizabeth Wirth Music, Trottier, Wong, Gelber Law Library, Chancellor Day Hall, Brown, Strathcona Anatomy and Dentistry, Stewart Biology, Burnside Hall, and Education for winter 2018.

Depend on conditions on information communication technology infrastructure, AMI, smart meters, software tools employ in electricity companies. They can make DSM / EE policies for each customer group using electricity, and group's customers will gradually expand over time.





In 2019, Hydro-Québec offers Dynamic pricing tools. The new dynamic electricity pricing design that will support consumers save costs by curtailing their power usage or moving some of their consumption to off-peak times. The new opportunities will need an online tool a self-serve savings simulator, a rate sign-up interface, a power usage tracker, and so on, which are currently in development.

Both services are intended for customers who pay Rate D and can reduce or replace the electricity consumption required by Hydro-Québec. For winter, 2019-2020, randomly selected residential and agricultural customers will be able to choose one of them, if they wish.

Table 1.3 Dynamic pricing tools apply for residential and agricultural customers show detail below [23].

Table 1.3 Comparison of Dynamic rate offerings with the base rate
 Taken from Hydro-Québec

www: <http://www.hydroquebec.com/business/customer-space/rates/dynamic-pricing.html>
 (Retrieved from website on February 27, 2020)

COMPARISON OF DYNAMIC RATE OFFERINGS WITH THE BASE RATE			
	RATE D (BASE RATE)	RATE D WITH WINTER CREDIT	RATE FLEX D
 Summer period*	Base rate price	Base rate price	Base rate price
 Winter period**, outside of critical peak events	Base rate price	Base rate price	Price lower than base rate
 Winter period**, during critical peak events (Max. 100 hours/winter)	Base rate price	Base rate price minus credit of 50¢ per kWh curtailed	Price higher than base rate: 50¢ per kWh consumed
 Impact on bill in relation to base rate	---	Potential for savings. Risk-free: the bill can only get smaller.	Potential for substantial savings. The bill can increase if consumption is not reduced during the critical peak events.

*Summer period: April 1 to November 30
 **Winter period: December 1 to March 31

1.6 Renewable resources integrate into the distribution grid and Smart Grid System

The National Energy Board (NEB or Council) announced its latest long-term energy outlook, Canada's Energy Future 2016: Energy Supply and Demand Forecasts by 2040 (EF 2016), in January 2016.

The report shows wind, solar, hydro, and the natural gas-fired generator that will be built to meet the demand for the increase and replace coal-fired power generators. Canada has many renewable energy resources. The hydropower generation remains a significant source of electricity in Canada. More than 80% of the electricity generated comes from non-fossil

energy sources in 2015, almost are hydroelectricity power. Figure 1-13 below is shown the Capacity Addition and retirements by 2040 in Canada.

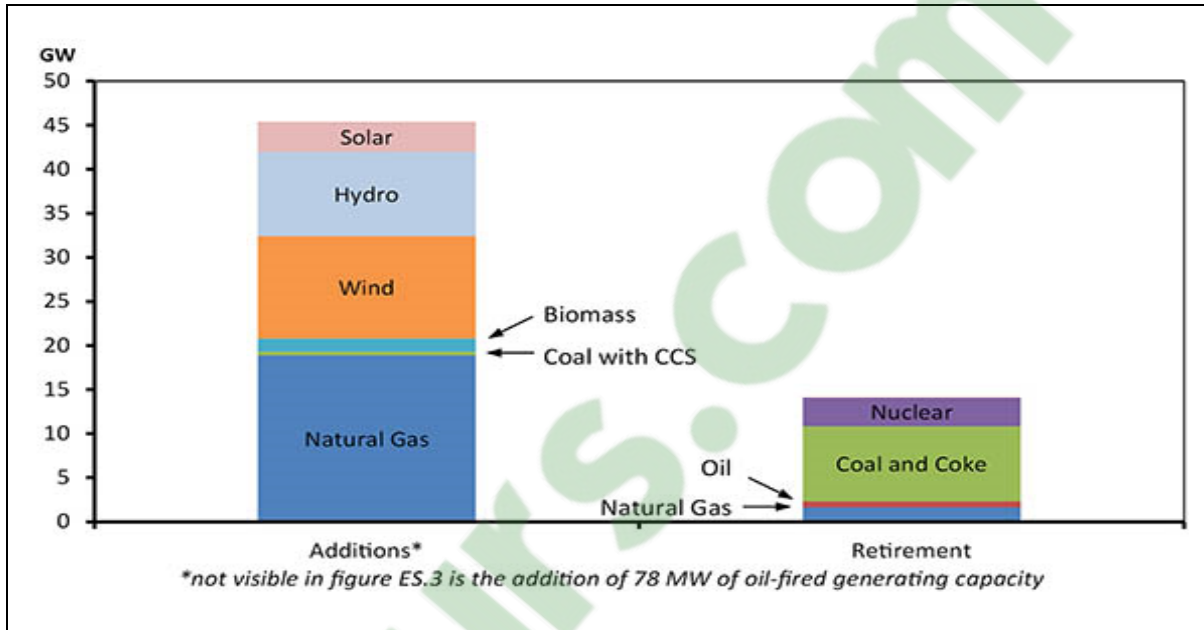


Figure 1.13 Generating Capacity Additions and Retirements by 2040, Reference Case
Taken from National Energy Board (2016, p. 9)

We also see in report the result of renewable generation development. Over the past decade, the generating capacity of renewable energy has increased rapidly. This trend is expected to continue as international organizations, governments, and industry associations plan for a broader deployment of renewable technologies globally.

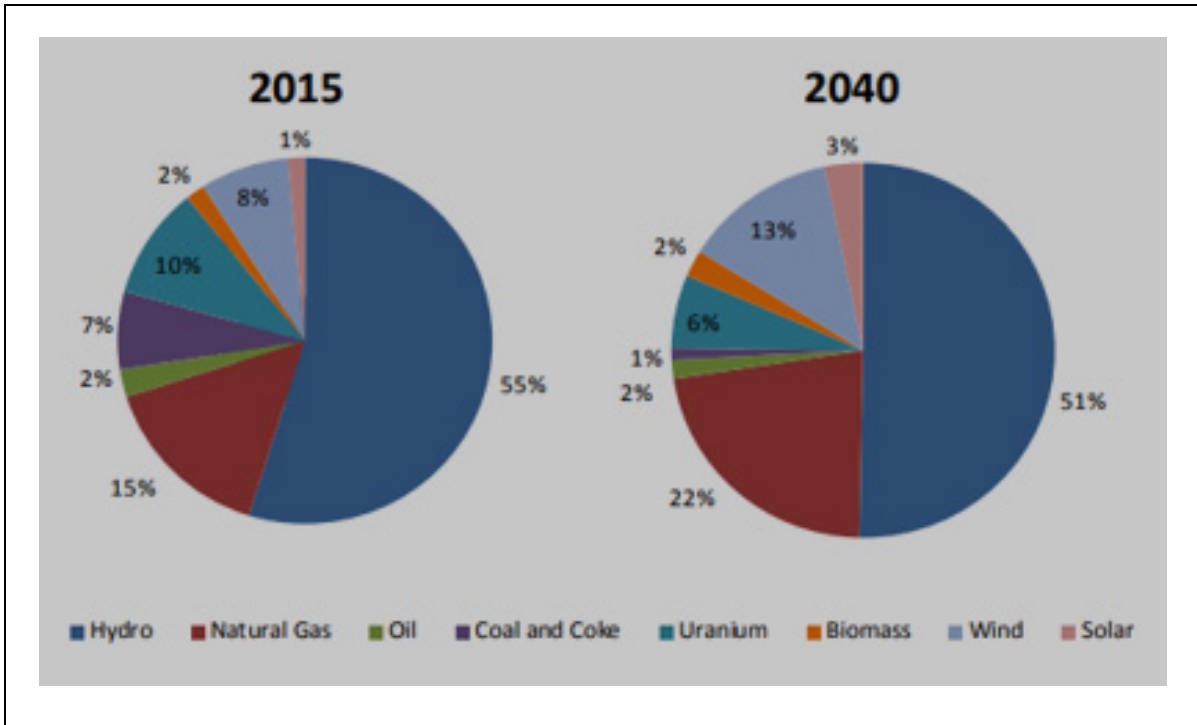


Figure 1.14 Capacity Mix by Primary Fuel, 2015 and 2040, Reference Case
Taken from National Energy Board (2016)

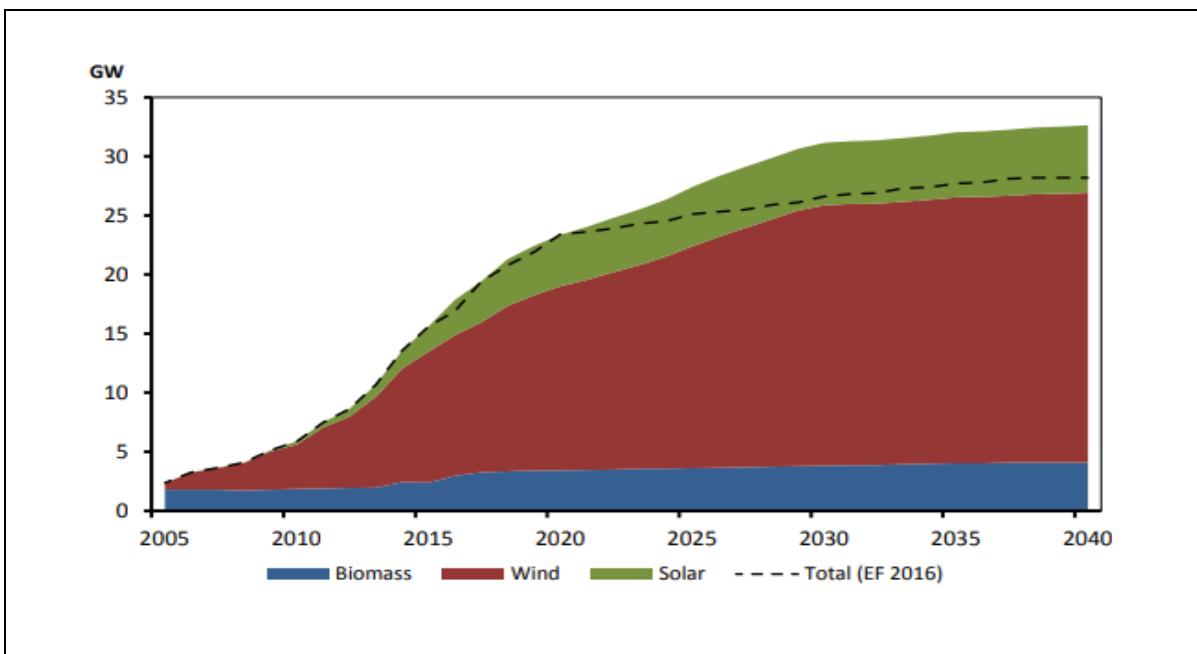


Figure 1.15 Non-Hydro Renewable Capacity, Reference Case
Taken from National Energy Board (2016, p. 7)

Besides, Canada exports a large amount of electricity to the United States, mainly hydroelectric power with storage. Hydroelectric power with storage also contributes to the growth of the integration and reliability of different types of renewable sources on both sides of the border. This position of strength was motivated by the collective and cooperative actions of the federal, provincial, and territorial governments, through numerous initiatives such as offset programs. The contracting through requests for proposals, standard incentives, and feed-in tariff programs, and ordered renewable portfolio criteria. As governments pursue policies to promote energy innovation and encourage the deployment and integration of renewable energy sources, Canada is dependent on electricity for its renewable energy sources. The creation, including hydropower, wind, solar, biomass, geothermal, and marine, will continue to increase. In the future, cooperation among jurisdictions will be needed to maintain Canada's renewable energy advantage. Combining federal, provincial, and territorial governments can have a significant impact in supporting the reliability and trade of electricity, improving the adequacy of the system and ensuring the sustainability of our electricity systems [24][25].

Nowadays, solar power is becoming more and more competitive to generate. Solar power gives an opportunity business for Hydro-Québec. Solar power could replace other energy resources like oil, coal, and natural gas early. Currently, Hydro-Québec has developed strategies that include the construction of solar generating stations.

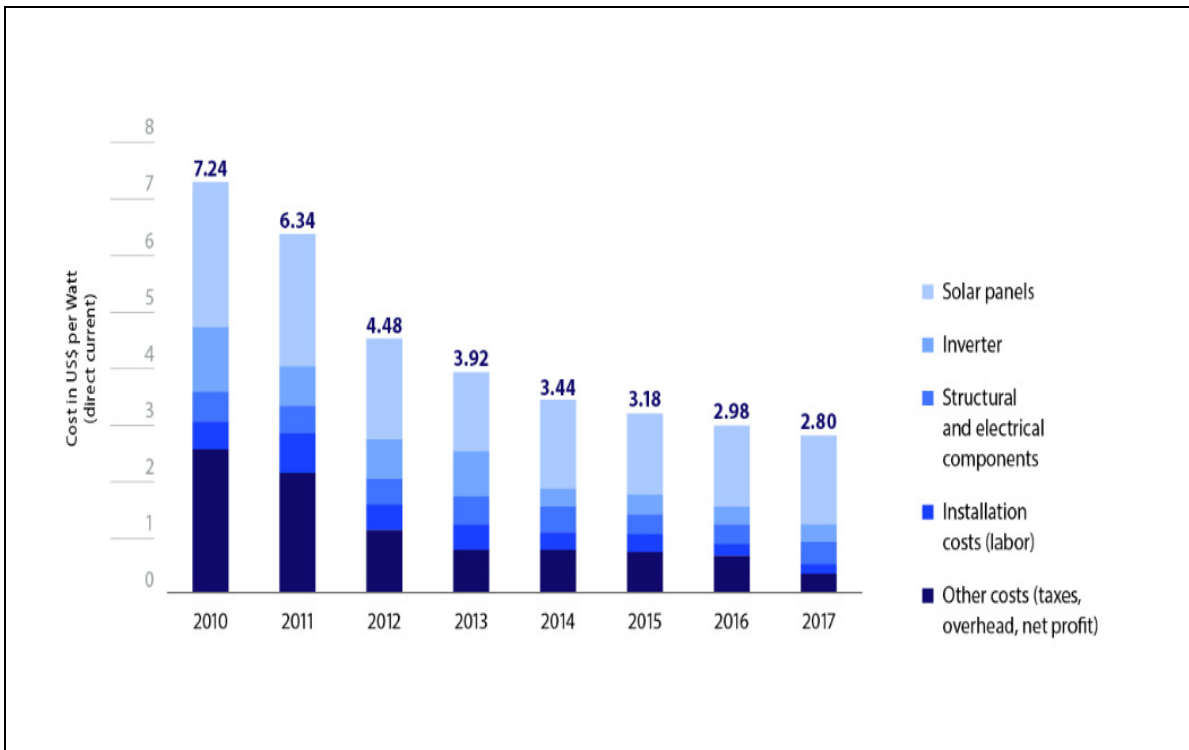


Figure 1.16 Changing Cost of Photovoltaic Solar Power Systems
U.S. Residential Market (2010–2017)

Taken from Hydro-Québec

www: <https://www.hydroquebec.com/sustainable-development/our-approach.html>
(Retrieved from website on February 27, 2020)

The energy transition brings additional opportunity to Hydro-Québec. For example, pursue acquisitions outside Québec, maximize energy export revenue, and increase power grid flexibility.

The market development of solar energy produced by photovoltaic resource could influence the operations of power companies on several levels below:

- 1) Reduced energy consumption.
- 2) Electric price changes.
- 3) Net Demand and Demand forecasting.
- 4) Supply and demand balance.
- 5) The rise in light load periods [26].

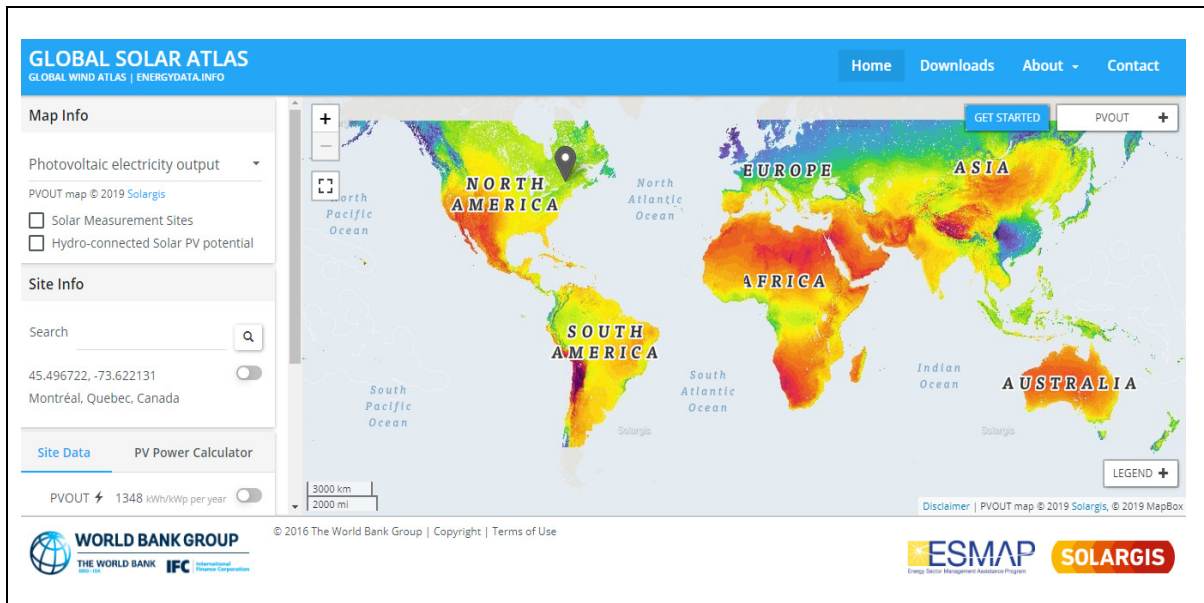


Figure 1.17 Montreal in Global Solar Atlas
 Taken from World Bank Group
 www: <https://globalsolaratlas.info/map>
 (Retrieved from website on February 27, 2020)

The one interesting was showed in the Global Solar Atlas that Montreal has more potential than some other city in China. Even though China is one of the biggest countries using solar energy.

In the case, New England, the New England ISO (Independent System Operator) observed a significant reduction in electricity demand during the heat wave in July 2018. The power capacity of the regional electricity system has reduced demand about 2,000 MW per day at peak around 1:00 pm. On a chilly, sunny day in April 2018, solar output raised a predicted record high at 1:00 pm. It reduces 2,300 MW the electricity demand on the regional electricity system. As a result, the New England electricity consumers use electricity at nighttime more daytime, because they get energy from the solar in the daytime. For the first time since at least 2000, Thanksgiving electricity demand was not highest in the morning when the New England electricity consumers had just turned on their ovens.

Traditionally, the independent system operators have relied on historical patterns of electricity usage to accurately predict the amount of electricity that must be generated to meet

real-time demand. However, as distributed solar power sources significantly change the pattern of electricity demand, making accurate predictions becomes challenging [27].

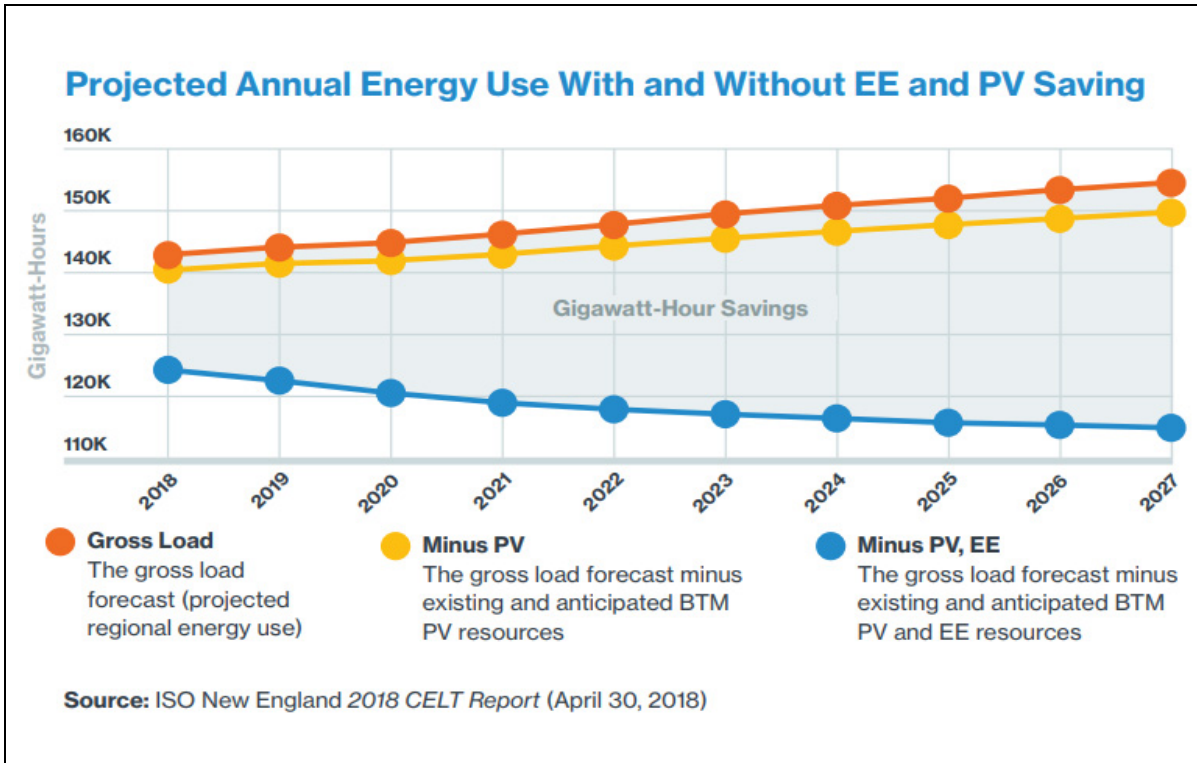


Figure 1.18 Projected Annual Energy Use with and without EE and PV Saving in New England Power Market
 Taken from ISO New England (2019)
 www: https://www.iso-ne.com/static-assets/documents/2019/03/2019_reo.pdf
 (Retrieved from website on February 27, 2020)



Figure 1.19 Solar energy effect to daily load profile

Taken from ISO New England (2019)

www: https://www.iso-ne.com/static-assets/documents/2019/03/2019_reo.pdf

(Retrieved from website on February 27, 2020)

Based on the above factors, forecasting the load of a building will depend on many factors that affect the choice of model and load forecasting method. The figure 1.20 below shows some factors effect to demand forecast.

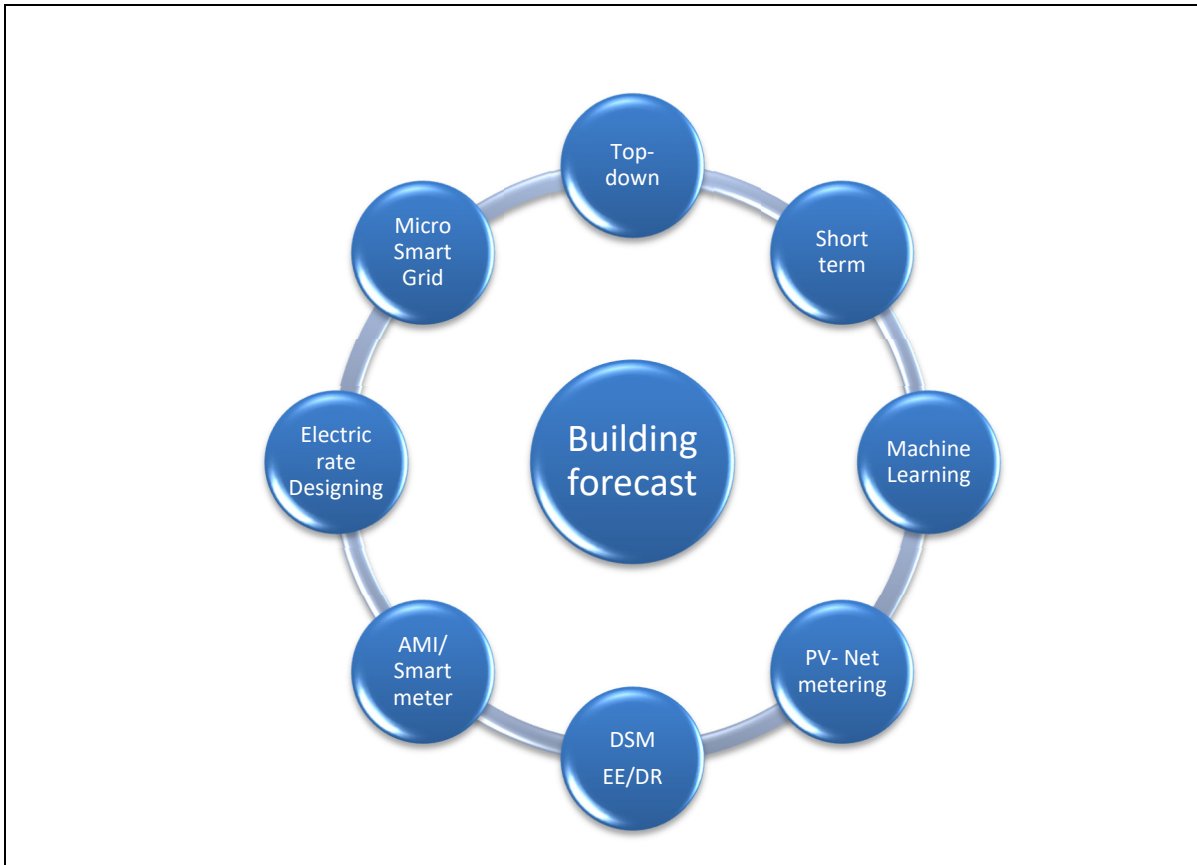


Figure 1.20 Overview building forecast and other factor effects

CHAPTER 2

LITERATURE REVIEW

2.1 Methodologies of electrical demand forecasting

Electricity demand forecasting has been used for numerous purposes. For example, the power demand forecast applies for investing in extending the power system, effective operation of the power grid, and establishing electricity rates. Moreover, they can apply for building energy efficiency policies, demand response, demand-side management programs. The electricity demand forecasting for the buildings is a challenging task. It massively depends on the behavior of the electric consumer, the technology of electrical equipment, and the uncertainty of the weather [1]. There is no individual forecast that can satisfy all the requirements of an electric company or an electricity user. Different forecasting tools will need to be used for different purposes.

Based on the time frame of the forecast, we could divide load forecasting problems into the following four groups following:

- 1) Very short-term load forecasting, forecasting horizon ranging from a few minutes ahead to a few hours ahead.
- 2) Short-term load forecasting, forecasting horizon ranging one day to two weeks ahead.
- 3) Medium-term load forecasting, forecasting horizon ranging two weeks to three years ahead.
- 4) Long-term load forecasting, forecasting horizon ranging three to twenty or fifty years ahead [1].

2.1.1 Methodology applying for electricity demand forecasting short term

As research articles published between 2005 and 2015, the article [3] was a research article concentrates mainly on the methods that are applied to electricity demand forecasting.

Medium-term and long-term electricity demand forecasting regularly will use methods related to econometric parameters. For short-term and very short-term electricity demand

forecasting purposes, use methods related to short term weather parameters and power demand historical data. Numerous short-term power demand forecasting methods are used, for example, analog dates, multiple regressions, time series, neural networks, fuzzy logic models, and expert systems and developed for the short-term forecast. The choice of short-term load forecast method depends not only on available data but also on the characteristics and detail level of the load forecast. Even, it could be used further than a particular method and then analyze the forecasts to choose the most reasonable method. Therefore, all electricity companies, electricity customers need to find the most suitable technique to use it.

According to this article, the trend of researchers using AI for load forecasting is increasing. However, researching using AI requires sufficient data and consideration to ensure computation with minimal time [3]. Most considerable forecasting methods are used statistical or artificial intelligence methods. The list of the fundamental method is shown below:

- 1) Artificial neural network models.
- 2) Fuzzy logic.
- 3) Time series models.
- 4) Grey (gray) prediction.
- 5) ARMA, ARIMA, SARIMA.
- 6) Regression models.
- 7) Support vector machines.
- 8) Genetic Algorithm.
- 9) Econometric models.
- 10) System dynamics models [19].

2.1.2 Linear Regression Models

Regression models have a long history of development. Many research articles have implemented both short-term load forecasts (STLF) and long-term load forecasts (LTLF). The report [1] shows that the regression analysis uses statistical methods, including the estimation of relations between input and output variables. Calendar and weather variables

are considered as independent variables, variables in the past load figures are dependent variables. The process of variables estimation depends on the knowledge and experience of the forecaster. The forecaster needs to predict the relationship of these input variables to the output variable. The authors [1] introduced a new method for applying regression research to STLF, in which interactions between weather and calendar variables are highlighted. Some special effects have been modeled using regression analysis, such as the variables that feature weekend effects and holiday effects.

Because the hourly electricity load will be a more detailed and more accurate reflection, the problem is how to get an hourly electricity load and a continually updated data. In the past, the hourly electricity load data could not be automatically and continuously collected because the modern metering system has not been installed. However, in recent years the new generation electric meter system (or Advanced metering infrastructure system) has been installed. This system can record, collect, store data, and transmit mostly online metering data (or hourly, daily). Therefore, nowadays, we have enough data to store, analyze, and forecast loads based on hourly data [28].

A challenging problem with this model is the need to use knowledge analysis, statistical methods to determine the relationship between input variables and outputs.

2.1.3 Artificial neural network models

ANN is a method that has been used a lot in recent years, which helped ANN become more and more applied in many fields. Some of the reasons ANN has been growing stronger are due to Big Data, the Internet of Things, and rapidly growing IT infrastructure. Algorithms and the structure of the network are continued research and development.

ANN is best known for being able to predict the output of nonlinear data sets. ANN can learn from input data to recognize the relationship between output and input variables. After the training process to find the parameters, the ANN was chosen which can make predictions of output values with corresponding input data. An ANN typically consists of at least one input layer and one output layer and one hidden layer between the input and output layers. This

layer is also known as the hidden layer. The deep neural network is the ANN has more than one hidden layer. In each layer, there are processing nodes (called neurons). Historical data can be used by ANN to train and create suitable neural networks. Use that neural network to make predictions with small predictive errors. ANN uses sum (with weighted and biased) methods and transforms functions (activation functions) to process data in each neuron. After training with numerous epochs, the error between the prediction and the actual value of the training function is small at an acceptable level, thereby determining the parameters of the neural network [3]. Chapters 3 and 4 in this thesis are presented ANN more detail. Figure 2.1 below shown the neural network and deep neural network:

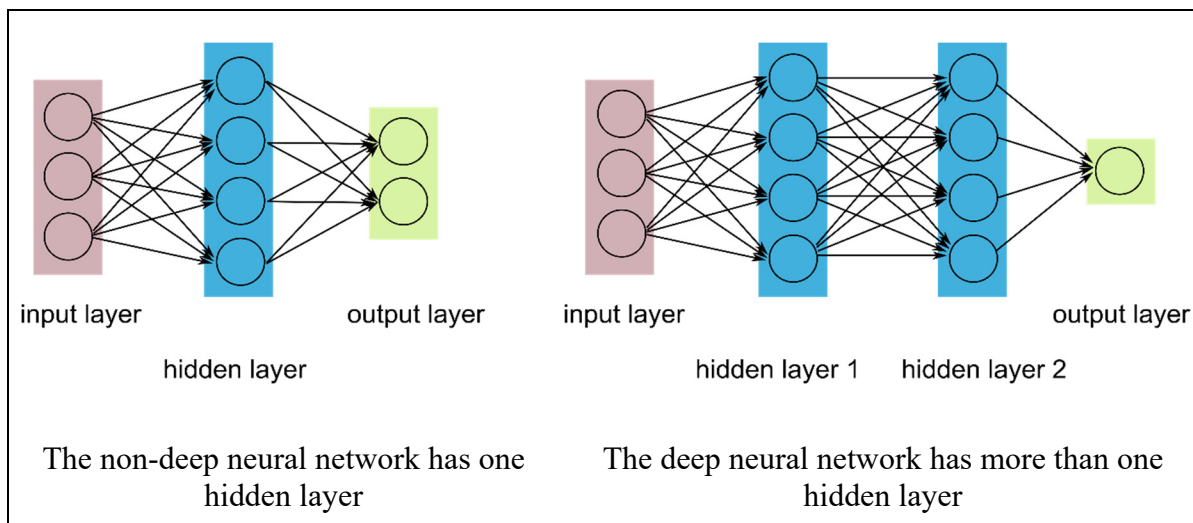


Figure 2.1 Non-deep and Deep feed-forward neural network

Adapted from Karpathy (2017)

www: <http://cs231n.github.io/neural-networks-1/>

(Retrieved from website on February 27, 2020)

2.1.4 Deep learning mode

Deep learning networks are considered neural networks with more than one hidden layer. Hidden layers also have a number of neurons. In recent years, many authors have studied this deep learning network structure experiments to develop applications. The authors of the paper [29] presented a new method of load forecasting using a deep learning network with a long short-term memory (LSTM) algorithm. The author used the same residential customer

data, applying two LSTM-based architectures presented: 1) standard LSTM architecture, and 2) LSTM based on S2S. The training and testing process are conducted with residential customer data with an hour and a minute resolution. The data is collected from the smart metering system. The results of the study show that the standard LSTM neural networks well in data in one-hour data while failing at one-minute resolution data. However, the S2S architecture works well on both data sets.

Moreover, the researchers found that the methods presented produce results comparable to other deep learning methods for predicting energy in academia. This article shows that the proposed neural network architecture can produce good results in training data sets. Furthermore, by increasing the number of classes and neurons in each class, it increases the learning ability of the neural network for the training data set. However, this neural network does not give good results for test data, which is known as the overfitting problem. The authors proposed a method of dropping out to reduce this problem. However, the authors also argued that the proposed algorithms needed to be tested on different real-world datasets in the future work. The proposed methods also need to be examined in the future with a new dataset in the real-world [29].

2.2 Input data

Each load forecast method would use input data to include in the forecasting model. Data types include historical electricity data, related time series parameters such as calendar, schedule, or weather-related information. Some critical factors of data should consider accuracy, reliability, quality, and availability. For traditional statistical methods as well as machine learning methods favor using these data.

2.2.1 Weather Variables

Weather is an essential factor that can affect the power consumption of electricity users, primarily residential areas where temperatures change considerably throughout the year. The main reason is that electricity users use air conditioning systems and heating systems. Therefore, many load forecasting methods have used weather information for short-term load

forecasting. Some weather variables are often used as dry bulb temperatures, such as wet bulb temperature, dew point temperature, relative humidity, wind speed and cloud cover, wind direction. Temperature (dry bulb temperature) is the most common of all the variables listed above and is widely used. There are many ways to accept temperature information in the model: previous hour temperature, current hour temperature, the difference between previous hour temperature and current temperature, maximum, minimum, or average temperature of the heat.

We have interpreted and modeled the relationship between load and temperature differently. However, for days with unusual weather, the electrical capacity increases significantly during hot or extremely cold summers due to the use of air conditioning or heating systems [3].

2.2.2 Calendar Variables

A year can be divided into four seasons and twelve months in a year. The months each year also have different load consumption, and there is a transition between months of the year, in some regions, the seasonal months. The month in winter and summer often have higher levels of electricity use than the spring and autumn months of the year. The days of the week also use different power, for example, office buildings may be closed on weekends, so the electricity consumption decreases lower on weekdays. On the morning of the weekend, people can wake up late, which changes the morning peak one or two hours later than the other working days. Each country usually shares the same weekend or holiday. However, due to different electrical customs that lead to different load consumption behavior, the grouping methods for classification may still be changed. Weekday types can always be different, even in the same country, is depend on areas with different uses of electricity, such as residential areas, office buildings, and industry zone. Therefore, the time variable has a significant influence on the level of electricity use [3].

2.2.3 Historical Demand Data

Traditional power meters (usually electromechanical meters) measure total power consumption and often do not have other information attached. For the new generation smart

meter, besides measuring power consumption, much other information is also measured and provided to the power supply unit and customers. Nowadays, the increasing use of smart grid technologies and smart metering systems have provided forecasters with detailed information on power consumption on smart meter systems. Besides the launch of advanced metering infrastructure (AMI), a large amount of consumption data is accessible. The AMI data has conducted a new wave of power demand forecasting. Power companies need to take complete advantage of AMI to be able to perform large-scale data analysis. The AMI system can convert metering data into useful information for power demand forecasting. By consumer sample availability, advanced analytical techniques allow power companies to extract detailed information on AMI data with a high level of cleaning, speed, and accuracy than before [1].

We need to consider some features when combining smart meter information to improving power demand forecasting models for better accuracy. Each electricity companies have progress on expanding the installation of smart meters into different stages. After the installation of smart meters brings some benefits to customers, and the behavior related to the use of smart meters by customers was changed. Customer's behavior who use smart meter not be similar to customers of traditional meter because the features and functions of the smart meters are different from traditional meters. Therefore, each type of consumer corresponding to installing meters require different forecasting models.

Usually, data for each 15-minutes interval with AMI systems are always available. Historical power consumption data created a data system for forecasters to develop load prediction models with more detailed resolutions. For traditional meter consumers, the developing forecast models for short intervals like 15 minutes is impossible due to the lack of data. Inside the smart grid system, the amount of data accessible is vast. Forecasters can confidently analyze this high volume of data and make accurate forecasts. In some cases, the data collected from the smart meter have common failures. We need to use more active pre-processing techniques to filter and fix these common failures before using them in predictive models [1].

The paper [30] shows that when using the AI method to forecast electrical loads in buildings, the selection of input data is significant. According to this paper, the input data be selected based on expert knowledge and data availability. Due to different experiment situations in different researches, different input characteristics have been decided as input data. The researchers divided them into three groups, namely meteorological, occupation, and all other groups based on the type of features input data. It can be seen that the majority of articles used meteorological data to make their forecast, accounting for over 60%. Meteorological data can be obtained from regional weather stations, including temperature, humidity, and other data like solar radiation, wind speed, and precipitation. However, just 29% of the articles evaluated used the information related to developing their forecast models. One of the main reasons because it is hard to get quality data.

The article also revealed that more than half of the authors used all other information to develop load prediction models. For example, historical electric data has been used along with indoor information, such as temperature and humidity. Furthermore, other information was shown various work schedules or seasonal patterns like an hour of the day, weekday type, holiday, and even activity calendar was utilized [17].

2.3 Neural network structure and optimal algorithms neural network

The authors [1] summarized that since 1990 ANN had been widely used for power demand forecasting. A part of AI is ANN, which can learn from historic data and does not require explicit physical system modeling. By only learning examples from historical data, a neural network is built on the relationship between input and output variables. In the case of building a load forecast model, the output is the amount of power consumption to be forecasted. After training, the neural network model is then used to predict. The researchers came up with several ANN network architectures used to predict power demand. For example, there is a feed-forward neuron backpropagation, Hopfield, Boltzmann machine. One of the most popular architectures is feed-forward backpropagation [1].

The artificial neural network is often called the most widely used machine learning model in building load forecasts [1]. Nonlinear and complex problems or the problem have difficulty identifying the parameters that can often be solved by the ANN model effectively.

Research by Neto and Fiorelli compared between a thermal model developed in Energy Plus and a simple ANN model, to forecast hourly power demand in a campus building. Mean Absolute Percentage Error (MAPE) of the ANN model is 10%, while in daily thermal models has an accuracy of over 12%. The study was also compared between a complex ANN model and a simpler ANN model. The result has shown that the complex ANN has average MAPE better 9.5% than simple ANN reaches [31].

Levenberg Marquardt model and the Bayesian Regularization model were presented in the paper [32]. The model illustrates the most valuable acting ANN architecture with the weight and bias in each neuron in the network. Both models use a standard backpropagation method to determine the more straightforward Jacobian matrix approximately the Hessian matrix. However, during the BR method, the neural network parameters are handled as random variables (Gaussian distribution). The parameters are also used in the optimization function are by Bayes' theorem [32].

2.4 Combining electricity demand forecast models

The researcher [33] showed that, compared with the individual forecast, the generally combined forecast showed better results. In 30 experiments compared, the combined forecast was about 12% MAPE better than an individual forecast. Under some perfect conditions, the combined forecast even reaches up to 20% MAPE better than an individual forecast. In other words, combined predictions are even better than the best single forecast. They are never less accurate than average single forecasts. Therefore combined forecasting is a suitable and effective method for forecasters [33].

The paper [30] presents forecasts of building power demand that can be organized into two methods: artificial intelligence methods and hybrid methods. The AI-based method utilizes past data to forecast power demand in the future. Technical and hybrid methods use thermodynamic equations to forecast power demand. AI-based approaches have become

popular in recent years because of the development of technologies, such as smart meter, big data, IoT, ANN, and deep learning algorithms. The paper also presented an in-depth evaluation of individual AI-based methods like neural networks, multivariate linear regression, and support vector machine. The paper also pointed out that synchronous prediction methods by combining different individual AI-based forecast models improve predictive accuracy [30].

CHAPTER 3

MACHINE LEARNING

3.1 Machine learning

A machine learning algorithm is an algorithm that is able to learn from data. However, what do we mean by Learning? Mitchell 1997 [34] provides a succinct definition “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E .”

Three concepts are Task, Performance, and Experience in Machine learning was discussed detail below:

3.1.1 The Task, T

For some tasks, it is too difficult to solve by writing programs, but with machine learning, it is possible to perform this task. Machine learning has grown very fast and has been used in many tasks.

In the definition above, learning is a way for us to achieve the capability to perform tasks. For example, to create an automation procedure require to create working robots, then walking will be a task. We can set up a walking robot program or programming a robot to learn how to walk. Machine learning tasks are often described in such a way that systems that machines can learn. They are usually represented as the vector $x \in R^n$, where each x_i input of the vector is another feature. For example, the features of an image are traditionally the pixel values of the image. Many types of tasks were solved by machine learning. Two types of popular machine learning tasks cover:

Classification: In this type of task, learning algorithms are often needed to create the function $f: R^n \rightarrow \{1, \dots, k\}$. When $y = f(x)$, the model assigns an input described by the vector x to a list defined by the number y . For example, the algorithm guesses the digits and letters on the envelope to determine the postal code. There are other variations of classification tasks, for

example, building the function f that results in a probability distribution each type of outputs. Some different types of tasks can be, for example, handwriting recognition, machine translation, etc.

Regression: In this task, from particular input data, learning algorithms are required to create the function $f: R^n \rightarrow R$ so that the relationship between the outputs data is usually the values. An example of an energy demand forecast is the task type. Forecasting future stock prices, predicting the number of passengers going on a train at Christmas, and New Year is a task [35][36].

3.1.2 The Performance Measure, P

The quantitative measurement of the performance of the evaluation is required to evaluate the effectiveness of a machine learning model. This P performance measure needs to be specifically designed for the T task system. For example, the tasks such as sorting and classifying, evaluate effectiveness performance of model are accuracy rate. We can create a comparable machine learning model by measuring the error rate. If the error rate is high, that means the model produces incorrect results. Typically, we consider the performance of machine learning algorithms on new data. The data has never been seen before because it determines its effectiveness if deployed in the real world. Therefore, we evaluate these performance measures by using a test data set separate from the data used to train and formulate machine learning systems [36].

3.1.3 The Experience, E

Different data sets will give different experience models. Data set characteristics will provide different models of experience. Because the training of machine learning models can be considered as experienced in the training data sets [36].

3.2 Machine learning algorithms groups

According to the learning method, Machine Learning algorithms are usually divided into four groups: Supervised Learning, Unsupervised Learning, Semi-supervised Learning, and

Reinforcement Learning. Several categories do not have Semi-supervised Learning or Reinforcement Learning.

3.2.1 Supervised Learning

Supervised Learning is an algorithm to predict the output (outcome) of new data (new input) based on known pairs (input, outcome). This data pair is also known as (data, label), instant (data, label). Supervised Learning is the most well-known group in Machine Learning algorithms.

The supervised learning algorithm is further broken down into two main categories:

3.2.1.1 Classification

A problem is called classification if the label of the input data is divided into a finite number of groups. For example, Gmail determines whether the email is spam or not. In the banks, the creditors determine if a customer can pay off the debt or not. The two examples above are divided into this category.

3.2.1.2 Regression

If the label is not divided into groups, but the label is a specific real value. For example, if we have data, we try to predict the price of a house has X m² has Y bedroom, and distance is the city center Z km. This problem is called regression.

3.2.2 Unsupervised Learning

In this algorithm, we do not know the outcome or label but only the input data. The unsupervised learning algorithm will depend on the structure of the data to perform a specific task, such as grouping (clustering). When we only have label data on x that do not know the corresponding y label, we will use Unsupervised Learning.

These algorithms are called Unsupervised Learning because, unlike Supervised Learning, we do not know the correct answers for each input data. Just like when we study, the teacher will not tell us that it is letter A or letter B. The unsupervised cluster is named in this case. Unsupervised learning problems are further broken down into two categories:

3.2.2.1 Clustering

An issue of grouping all x data groups into small groups based on the association between the data in each group.

For example, we can group customers based on purchasing behavior. Clustering is like giving children lots of pieces with different shapes and colors, for example, triangles, squares, circles with blue and red, then asking them to divide them into groups. Although it is not possible for children to know which pieces correspond to which picture or color, it is still possible to classify puzzle pieces by color or shape.

3.2.2.2 Association

It is a problem when we want to discover a rule based on a lot of given data. For example, male customers who buy clothes often tend to buy more watches, sunglasses, or belts. The person who was watching Spider-Man movie has a trend to watch more Bat Man movies. The Recommendation system creates based on customer behavior that can promote shopping demands.

3.2.3 Semi-Supervised Learning

The issue of this group is the rest of the two groups mentioned above. When we have an enormous number of X data, but just a part of them has labeled as Semi-Supervised Learning.

A typical example of this group is only a portion of a photo or text labeled. For example, a picture of a person or animal and writing of scientific or political are labeled.

Many Machine Learning problems belong to this group because collecting data with labels is very time consuming and has high costs. Many types of data even require a new specialist to be labeled (medical images, for example). In contrast, unlabeled data can be collected at low cost from the internet.

3.2.4 Reinforcement Learning

Reinforcement learning is a problem that helps a system automatically determine context-based behavior to achieve maximum benefit (maximizing the performance).

Currently, Intensive Learning is mainly applied to game theory, and the algorithm must determine the next step to obtain the highest score [35].

Other point of view, we can see five questions for data science applies machine learning algorithm from; <https://docs.microsoft.com/en-us/azure/machine-learning/studio/data-science-for-beginners-the-5-questions-data-science-answers>

- 1) This is A or B?
Classification Algorithm or Multi-classification Algorithm.
- 2) Is this Weird?
Anomaly Detection Algorithm.
- 3) How many or how much?
Regression Algorithm.
- 4) How is this organized?
Cluster Algorithm.
- 5) What should I do now?
Reinforcement Learning Algorithm.

3.3 Capacity, Overfitting, and Underfitting

The main difficulty of machine learning is that our algorithm needs to work adequately on new data never seen before, not only on the data used for model training. The machine learning model can execute well on before unobserved input data is generalized.

Usually, we have a training model after training. On each learning data set, we can build a model and calculate and measure efficiency through the error function, called learning error or training error. Then we need to reduce this error.

When we use a machine learning algorithm, the parameters were does not modify before finding the final model. The sample input will divide into two sets of data (training and test data). We use the sampled training set, train the model to find the parameters of the model to reduce the training set error. Next, we use the model to check with the test set. According to this procedure, the proposed test error is equal to or higher than the value of the training error. The test error value determines the capacity of the machine learning models can work in the real world.

- 1) Create small training errors.
- 2) Minimize the gap within testing error (generalization error) and training error.

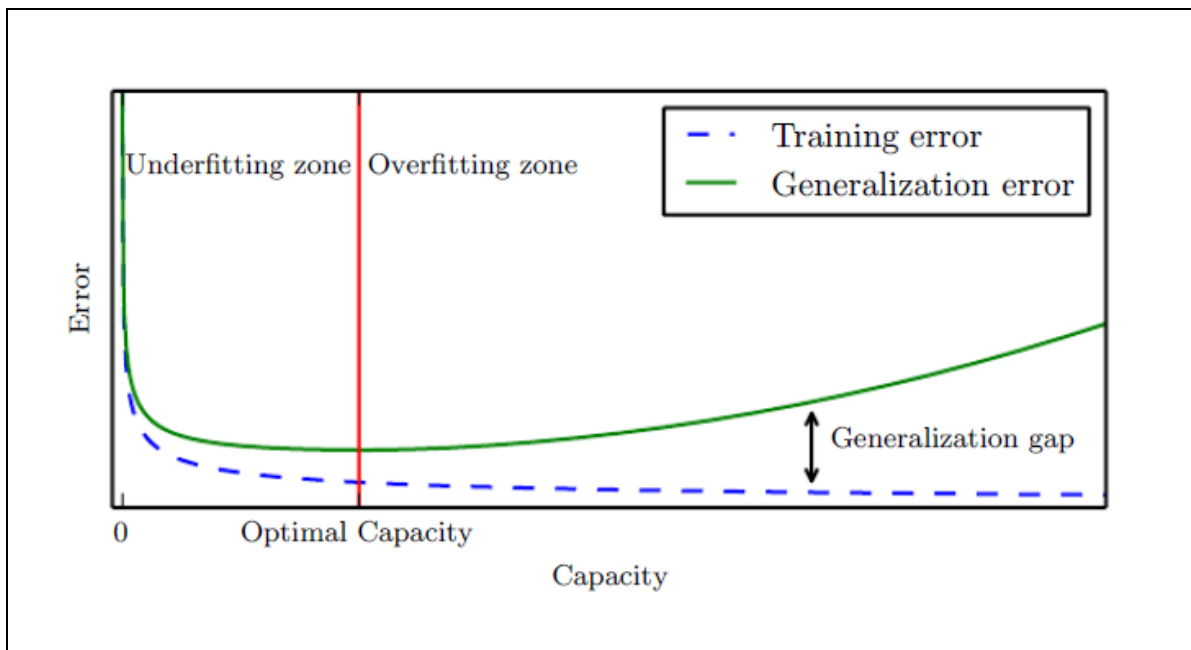


Figure 3.1 Underfitting and Overfitting
Taken from I. Goodfellow et al (2016, p.113)

Overfitting and underfitting are the two main challenges of machine learning. The underfitting occurs when the machine learning model cannot achieve an adequately low error

value on the training data set. The overfitting happens when the training error and the test error have a big gap.

Low capacity models may have difficulty to learn from training data. Strong power models may be appropriate in remembering the characteristics of training data, but that does not work well on test data.

One way to improve the learning ability of the algorithm is to choose appropriate mathematical models [36].

3.4 Methods to avoid overfitting

Overfitting is a common unwanted phenomenon. Machine Learning modelers need to know the techniques to prevent this phenomenon. Overfitting is the phenomenon of finding the model is too fit with training data. Still, this model no longer fits on test data. Overfitting happens when the model is too complicated to simulate training data. Overfitting happens, especially when the amount of training data is too small while the complexity of the model is too high. In the above example, the complexity of the model can be considered the order of the polynomial to be searched. In Multi-Layer Perceptron (MLP), the model's complexity can be regarded as the number of hidden layers and the number of units in hidden layers. A good model if we have both train error and the test error is low.

Some method such as Validation, Early Stopping Regularization, and Drop out was used to avoid overfitting problem.

3.4.1 Validation and Cross-validation

We are still used to dividing the data set into two small volumes of training data and test data. In addition, one thing we always want to note is that when building models, we cannot use test data. The simplest method is to extract the training data set into a small subset and perform a model review on this subset. The small data set extracted from this training set is called a validation set. At this time, the training set is the rest of the original training set.

Train error is calculated on this new training set, and there is another concept defined similarly to the validation error. In other words, the error is calculated on validation.

With this new concept, we find the model so that both train error and validation error are small, thus hope to predict that the error test is small. The commonly used method is to use many different models. The model for the smallest validation error will be a good model.

Usually, we start from a simple model, and then gradually increase the complexity one. When the validation error tends to increase, select the model immediately before. Note that the more complex the model is, the smaller the training error to be.

Cross-validation is an improvement of validation, with the amount of validation data being small. However, the quality of the model is evaluated on many different validation files. A common way to use is to divide training set to k subsets without a common element, with nearly equal size. At each test, one of the subsets is taken as a validation set.

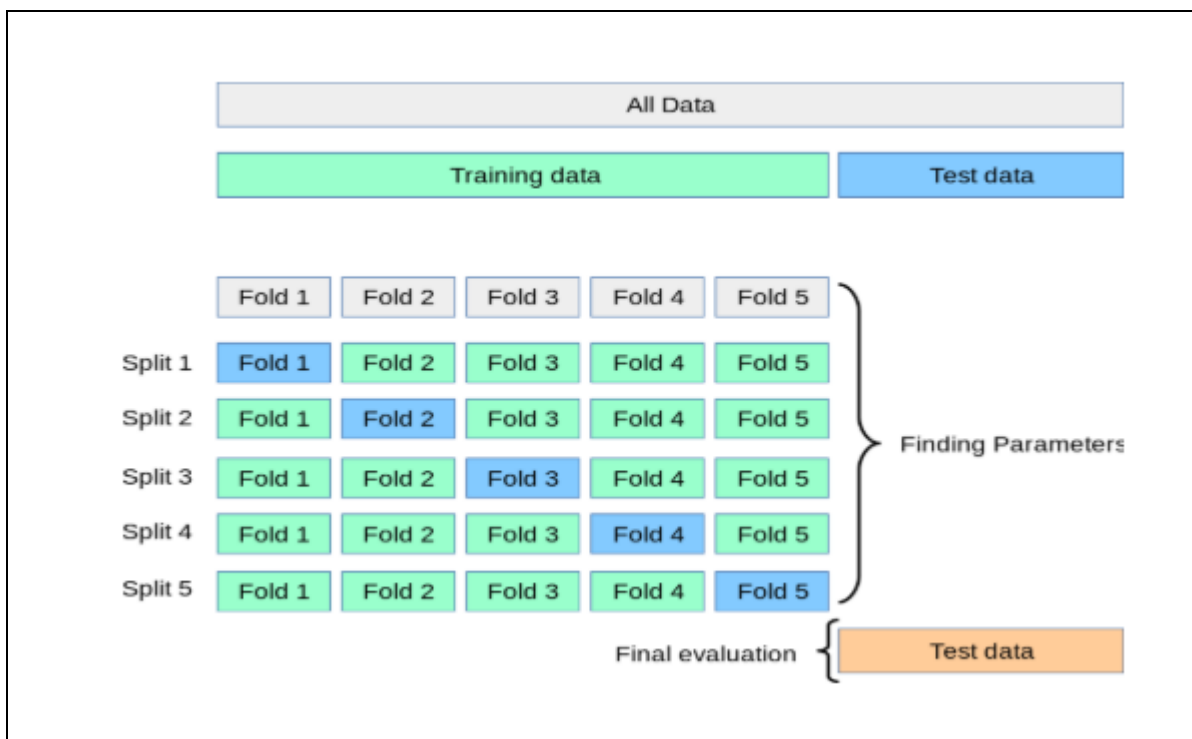


Figure 3.2 Cross-validation with k fold

Taken from Pedregosa et al (2011)

www: https://scikit-learn.org/stable/modules/cross_validation.html

(Retrieved from website on February 27, 2020)

Note that if we do a lot of hyperparameter tuning, our system ends up fine-tuned to perform well on the validation data. However, this system will likely not perform as well on unknown datasets (test set). The performance will usually be slightly worse than what we measured using cross-validation. When this happens, we must resist the temptation to tweak the hyperparameters to make the numbers look good on the test set; the improvements would be unlikely to generalize to new data [35][36][37][38].

3.4.2 Early Stopping

When we apply Machine Learning to solve many problems, we need to use algorithms to find solutions. For example, it is the Gradient Descent algorithm. In general, the loss function decreases as the number of epochs increases. Early stopping stopped the algorithm before the loss function was too small, helping to avoid overfitting.

A commonly used technique is to split from the training set to a set of validation as above. After one a number epoch, for example, 1000 epoch, we count both train error and validation error until we can see the validation error tends to increase. Then we stop training and return to using the model corresponding to the point, and error validation reaches small value.

3.4.3 Regularization

This technique will add a part noise to the loss function, considered learning with an unreliable teacher. In this case, ANN will not identify or simulate too high accurately with the training data set, just good enough. Therefore it is a method to avoid over-fitting problems. The most common regularization technique is adding a loss function to sum-squared weights. This technique is often used to evaluate the complicated model. The bigger the number of terms, the more complicated the model is. This new loss function is often called a regularized loss function.

For example, the linear regression loss function shows below:

$$L(w) = \frac{1}{2N} \sum_{i=0}^N (y - w_i x)^2 \quad (3.1)$$

Moreover, the regularized linear loss function shows below:

$$L(w) = \frac{1}{2N} \sum_{i=0}^N ((y - w_i x)^2 + \lambda w_i^2) \quad (3.2)$$

λ : Regularization parameter, $\lambda > 0$ and is small value.

This technique was detail explained by paper A Simple Weight Decay Can Improve Generalization [39].

3.4.4 Dropout

The robust machine learning systems are deep neural networks with a large number of parameters. However, a neural network with many parameters often has a problem overfitting serious. The number of parameters in the significant networks is also slow to training and executing it more challenging to solve with overfitting. Dropout is a technique for solving this obstacle. The key idea is a method of randomly shutting down neurons in networks. Turn off instant for neurons with zero value and calculate feedforward and normal backpropagation during training. Drop out technique not only reduces over-fitting but also reduces the amount of computation.

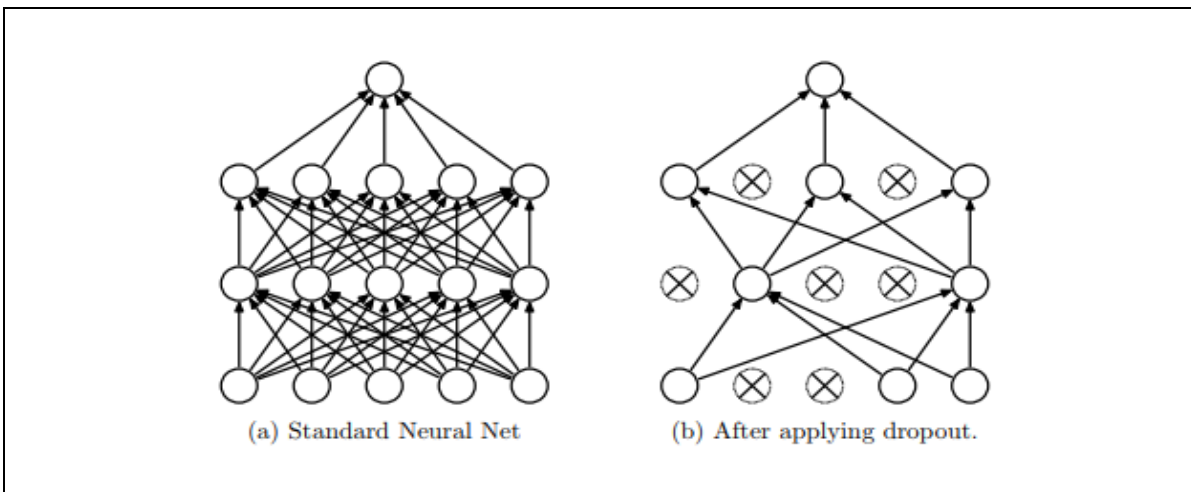


Figure 3.3 Dropout Neural Net Model
Taken from Srivastava et al (2014)

In figure 3-3 Left: A standard neural net has two hidden layers. In figure 3-3 Right: An example of a thinned neural network produced by applying dropout to the network on the left. Crossed units have been dropped [40].

Dropout is a method that applies to deep learning models with the number of model parameters in neural networks. This network often solves problems with large amounts of data and requires complex models [41].

3.5 Model parameters and Loss function

3.5.1 Model parameters and Loss function

Each machine learning model was described by model parameters. The task of a machine learning algorithm is to find the model parameters that are appropriate for each problem. Finding model parameters are closely related to performance (evaluations). Our goal is to find the model parameters so that the assessments have the best results. For example, in the regression problem, the best result is when the difference between the predicted output and the real output is the least.

The relationship between a rating and model parameters is often described through a function called a loss function (loss function or cost function). Loss functions are usually of small value when the evaluation gives good results and vice versa. Finding model parameters that allow the evaluation to return useful results is equivalent to at least the loss function. Therefore, building a machine learning model is to solve an optimization problem. That process can be considered as a learning process of the machine. The process of solving the optimization problem will be done with the training set and validation set (dev set) and evaluated through the test set. Therefore, the final aim is to find the model to respond to the test set, not just the solution to determine the smallest optimal function in the training set.

The set of model parameters is usually denoted by θ ; the loss function of the signed model is equal to $L(\theta)$. The problem of minimizing loss function to find model parameters is often written in the form of:

$$\theta^* = \arg \min_{\theta} L(\theta) \quad (3.3)$$

$\arg \min_{\theta} L(\theta)$ is interpreted as the value θ^* so that the function $L(\theta)$ reaches the minimum value. When using the function $\arg \min_{\theta} L(\theta)$, we must specify which variables it is executed by writing the variables below min (here θ). To understand machine learning algorithms, we essential to understand the optimal techniques.

There are some examples for the Loss function:

Linear Loss function shows below:

Where $\hat{y} = \sum_{i=1}^N w_i x_i + b = \sum_{i=0}^N w_i x_i$ with $x_0 = 1, w_0 = b$

Linear Loss function can be rewritten as:

$$L(w) = \frac{1}{2N} \sum_{i=0}^N (y_i - \hat{y})^2 = \frac{1}{2N} \sum_{i=0}^N (y_i - w_i x_i)^2 \quad (3.4)$$

In this case, linear loss function $J(\theta) = L(w_i, b)$, model parameters θ are w_i and b . With the training data set (x_i, y_i) , we need to find model parameters w^* to the problem of minimizing linear loss function in the form:

$$w^* = \arg \min_w L(w, x, y) \quad (3.5)$$

Logistics Loss function can be rewritten as:

$$L(w) = - \sum_{i=0}^N (y_i \ln z_i + (1 - y_i) \ln(1 - z_i)) \quad (3.6)$$

Where $z(w) = f(w_i x)$.

3.5.2 A simple machine learning formula

According to Slide in the course [Machine Learning Department School of Computer Science Carnegie Mellon University], they show that we can consider a simple formula for machine learning as follows:

1) Given training data:

$$\{x_i, y_i\}_{i=1}^N \quad (3.7)$$

2) Choose each of these

- Decision function:

$$\hat{y} = f_{\theta}(x_i) \quad (3.8)$$

- Loss function:

$$L(\hat{y}, y_i) \in R \quad (3.9)$$

3) Define the goal

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^N L(\hat{y}, y_i) = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^N L(f_{\theta}(x_i), y_i) \quad (3.10)$$

4) Train with stochastic gradient descent (take small steps opposite the gradient)

$$\theta_{t+1} = \theta_t - \eta_t \nabla_{\theta} L(f_{\theta}(x_i), y_i) \quad (3.11)$$

η_t : learning rate

We can explain 4 step following:

Step 1: We will select the input training data (x_i) corresponding to the output training data (y_i). We have an N sample (x_i, y_i).

Step 2: Find the decision function or the neural network model with the parameters θ (delta) and the input data x_i . We will select the loss function to have the smallest errors between decision function and the given output data y_i .

Step 3: Find the parameter delta to satisfy the condition of the minimum loss function's argument.

Step 4: Implement the stochastic gradient descent method to figure out the optimal parameter delta. Then, we can determine the decision function (or neural network model) with the optimal parameter. Then we will use the new input data with the decision function to forecast the new output.

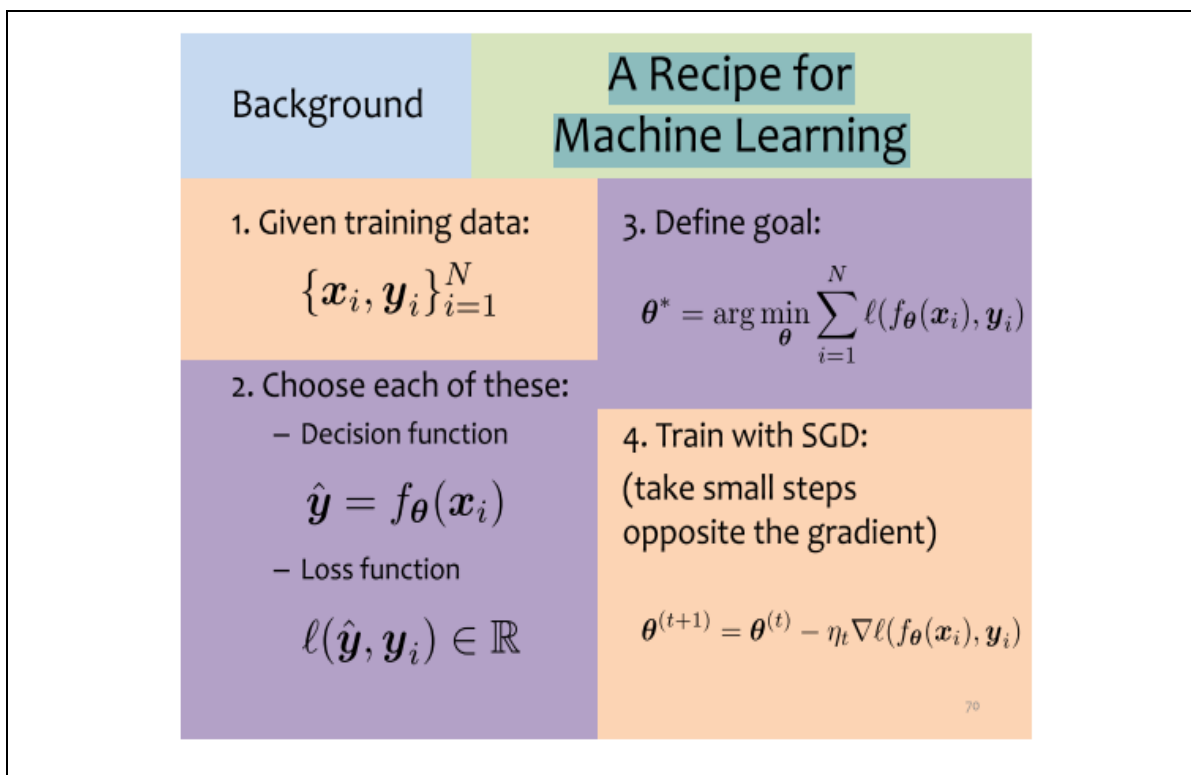


Figure 3.4 A Recipe for Machine Learning

Taken from Gormley (2018)

www: <http://www.cs.cmu.edu/~mgormley/courses/10601-s18/slides/lecture11-nn.pdf>

(Retrieved from website on February 27, 2020)

Stochastics gradient descent is a popular method to find model parameters, almost method applying gradient with a first-order gradient or second-order gradient.

3.6 Feature engineering

Data points are sometimes measured with different units; for example, we can measure distance by meter or feet. Alternatively, there are two variables (of the data vector), the difference is too big, for example, one variable has a value between 0 and 1000, and the other variable has a value of 0 to 1. At this point, we need to standardize the data before taking the next steps.

Data standardization is one part of the pre-processing data work. Standardization does not focus on the loss, missing, outlier and abnormal data.

Its primary task is to process the training data vector with the same dimensions. If there are different dimensions, it is necessary to identify and uniformly synchronize these vectors before training the ANN network.

The vectors are calculated and converted to the corresponding magnitude, which facilitates the Loss function's minimum finding.

When training, it is necessary to standardize data on vectors to reduce the calculation volume, while still finding relevant input data and keeping the relationship between the input and output data.

Several standardized methods are used:

3.6.1 Rescaling

The easiest method is to take all the components to the same range $[0,1]$ or $[-1,1]$. For example, depending on the application. If we want to include a component (features) about $[0,1]$, the formula will be:

$$x_{new} = \frac{x - \min(x_i)}{\max(x_i) - \min(x_i)} \quad (3.12)$$

If we want to include a component (feature) about $[-1,1]$, the formula will be:

$$x_{new} = \frac{x - 0.5(\max(x_i) + \min(x_i))}{\max(x_i) - \min(x_i)} \quad (3.13)$$

Inside x is the initial value, x_{new} is the value new after standardization $\min(x)$, $\max(x)$ calculated on all training data in the same component. The formula was done on each element of the data vector x .

3.6.2 Standardization

Another commonly used method is to assume that each component has a normal distribution with mean (an expectation) is 0, and a variance is 1. Then, the standardized formula will be:

$$x_{new} = \frac{x - \mu}{\sigma} \quad (3.14)$$

μ , σ are the mean and variance (standard deviation) of the component on all training data.

3.6.3 Scaling to unit length

Another widely used option is to standardize the components of each data vector so that the entire vector has a magnitude (Euclid, norm 2) equal to 1. The formula will be:

$$x_{new} = \frac{x}{\|x\|_2} \quad (3.15)$$

3.7 Linear Regression

Linear regression is an example of a simple machine learning algorithm. As the title suggests, the regression problem is proposed to be solved by linear regression. In other words, the built-in system can take an $x \in \mathbb{R}^n$ vector as input and predict the value of a scalar $y \in \mathbb{R}$ as its

output. The output of linear regression is a linear function of the input variable. With an input value vector, through the predicted model, a value of \hat{y} is obtained. We define the output as:

$$\hat{y} = \sum_{i=1}^N w_i x_i + b = \sum_{i=0}^N w_i x_i \quad (3.16)$$

with $x_0 = 1, w_0 = b$

$$\hat{y} = \mathbf{w}^T \mathbf{x} + b \quad (3.17)$$

Where $\mathbf{w} \in \mathbb{R}^n$ is a vector of parameters.

Parameters are values w that affect linear function. In this case, the fact that we multiply through the loss function before summing the contribution of all characteristics. We can act as a set of weights that determine the impact of all features on those predictions. If the features x is given a positive weight, the increase in this prediction value will increase. When the weight receives a negative, the value is reduced. When the value weight is high, it has a significant predictive effect. If the weight is 0, no prediction is available.

Therefore, we have a definition of T (Task): predict \hat{y} from x by transporting $\hat{y} = \mathbf{w}^T \mathbf{x}$.

Then we need a determination of P (Performance) measurements.

Assume we have a purpose matrix that includes input examples that we will not use for training, to evaluate the model's efficiency. We also have a target regression vector that gives an actual y value for each of these examples. Since this data set will only be used for evaluation, we call it a testing package. We call on input matrices like $\mathbf{x}^{(\text{test})}$ and regression target vectors as y . One way to measure model performance is to calculate the average squared error of a model at a set. If $\hat{y}^{(\text{test})}$ gives the predictions of the model on the test set, then the loss function mean squared error is presented by

$$L(\mathbf{w}) = \frac{1}{2N} \sum_{i=0}^N (w_i x_i - y_i)^2 \quad (3.18)$$

$$L^{test}(w) = \frac{1}{2N} \sum_{i=0}^N (\hat{y}^{(test)} - y^{(test)})_i^2 \quad (3.19)$$

Intuitively, the error measurement was dropped to 0 when $\hat{y}^{(test)} = y^{(test)}$. The error increases each time the Euclidean distance between forecasts and objectives increases.

$$L^{test}(w) = \frac{1}{2N} \sum_{i=0}^N \|\hat{y}^{(test)} - y^{(test)}\|_2^2 \quad (3.20)$$

To implement machine learning algorithms, we need to create an algorithm to change the weight by reducing the L^{test} . The algorithm can learn from experience by observing the training set $(\mathbf{x}^{(train)}, \mathbf{y}^{(train)})$. One way to do this simple is to minimize the mean squared error on the training, L^{train} .

To minimize L^{train} , we can solve it is with first-order gradient function equal to 0

$$L^{train}(w) = \frac{1}{2N} \sum_{i=0}^N \|\hat{y}^{(train)} - y^{(train)}\|_2^2 \quad (3.21)$$

$$\frac{\partial L^{train}(w)}{\partial w} = 2(\mathbf{w}\mathbf{X}^{(train)} - \mathbf{y}^{(train)})(\mathbf{X}^{(train)}) = 0 \quad (3.22)$$

$$\mathbf{w} = (\mathbf{X}^{(train)T}\mathbf{X}^{(train)})^{-1}\mathbf{X}^{(train)T}\mathbf{y}^{(train)} \quad (3.23)$$

The equation system with the solution given in equation (3.23) is called the standard equation. The evaluation of equation (3.23) is a simple learning algorithm. For an example of an active linear regression algorithm, see Figure 3.5.

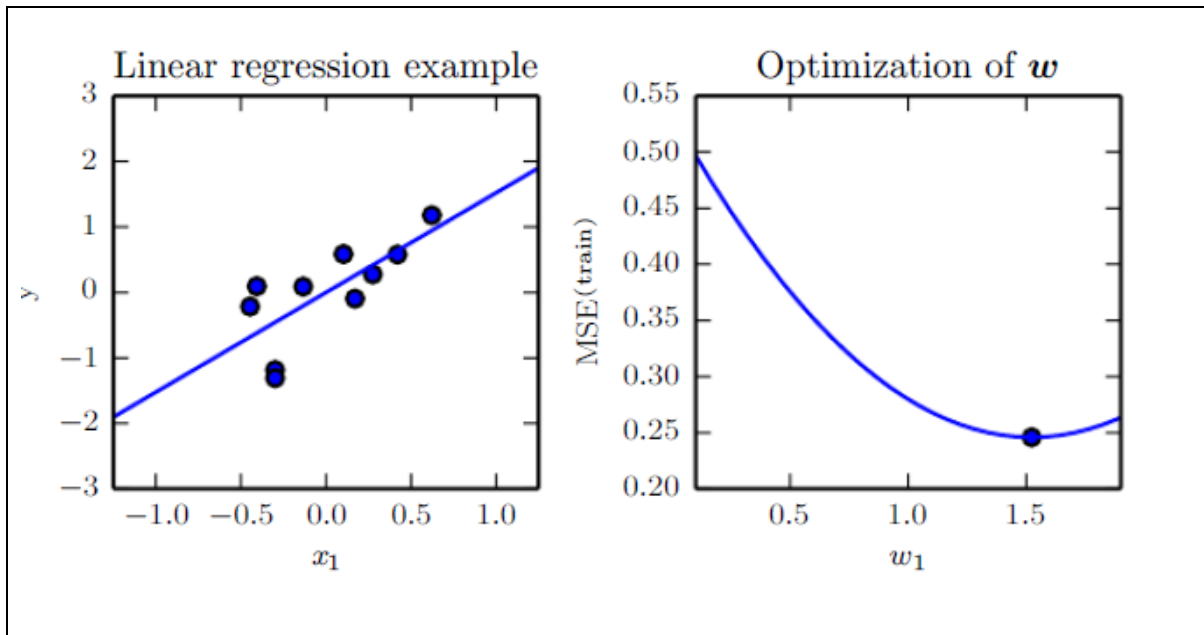


Figure 3.5 A linear regression problem
Taken from I. Goodfellow et al (2016, p.107)

Figure 3.5 shows a linear regression problem, including ten data values, each containing a characteristic in the learning data set. Because there is only one feature, the weight vector w receives only one parameter, w_1 . (Left picture).

Note that the linear regression teaches how to adjust w_1 so that the straight-line $y = w_1x$ is as nearby as possible to the path of all training datasets. The minimize value represents the value of w_1 found by standard equations, which reduces the mean square error on the training set. (Right picture).

It is worth to note that, the term linear regression is often used to refer a slightly more complex model with an additional parameter that is a bias b in this model.

$$\hat{y} = \mathbf{w}^T \mathbf{x} + b \quad (3.24)$$

Therefore, we find the parameters on the prediction of a linear function. This addition for affine functions proposes that the sample prediction graph always seems like a line, but it doesn't have to go through the point of origin (0,0). Instead, the biased parameter b was

added, we can continue to use the weighted model but increasing x with the additional one input. The parameter bias b weight is corresponding to an additional entry.

The bias parameter specific to the output will be b when the input values are zero. Under certain conditions, the outputs will be results with probability rather than a value specifically. For example, it is possible to forecast the temperature every hour of the next day, however forecast the probability of precipitation during that time. Linear regression is the algorithm given as a good example to explain machine learning algorithms.

The linear regression model is shown as a polynomial first-degree according to the equation below.

$$\hat{y} = \mathbf{w}^T \mathbf{x} + \mathbf{b} \quad (3.25)$$

By introducing x_2 as another feature given for the linear regression model, we can explore a quadratic model as a function of x_2 :

$$\hat{y} = w_1 x_1 + w_2 x_2^2 + b \quad (3.26)$$

Although this model fulfills the quadratic function of its input. But output remains a linear function of the parameters (w_1, w_2) so that we can still use the standard equations to solve. We also can attach more x powers like further features, for example, to get n -level polynomials:

$$\hat{y} = b + \sum_{i=1}^n w_i x_i^n \quad (3.27)$$

For machine learning algorithms to work accurately, it depends on a large enough amount of input and the complexity of the task to be implemented. Some simple models will be ineffective when solving complex tasks. A power model can solve complex problems; however, using a power model to solve simple problems will often lead to an overfitting problem.

Function Linear Regression $\hat{y} = f(w,x)$ is a linear function followed by w and x . In reality, Linear Regression can be applied to models that only need linearity w parameter. For example:

$$\hat{y} = w_1x_1 + w_2x_2^2 + w_3\sin(2\pi x_1) + w_4\cos(x_1x_2) + b \quad (3.28)$$

Although in the above section, we see that this method can be applied if the relationship between output (y) and the input (x) is not necessarily linear. The most challenging problem with prediction is determining the form of the function. It takes a lot of analysis and expert knowledge to explain the function form. The researcher must accept the assumption that the function form is true and then find its parameter. The parameters will be linear. In many cases, it is very difficult to determine the type of function in the real world. While using ANN, with enough data set, the model will learn from the data and establish ANN format after the training process [35].

The next chapter will mention the solution to this problem.

CHAPTER 4

NEURAL NETWORK

This chapter will discuss in more detail the neural network model. The neural network is one model that was popular in recent years in machine learning. In this chapter, the main subjects of the neural network, some concepts, such as neural network design, the ability of the neural network, objective function, the structure of neural layers, weight and bias, activation functions, were introduced. Then several techniques to limit overfitting during neural network training and the training algorithms were presented (first-order and second-order gradient descent algorithm). Forward and backpropagation derivative methods were mentioned at the end of the chapter.

4.1 The Perceptron Learning Algorithm

The Perceptron Learning Algorithm (PLA) is a Classification algorithm. For the simplest case: there are only two classes (classes) with only two classes called binary classification and only work in a particular case. The function that defines the Perceptron class label (x) = $\text{sgn}(w^T x)$ can be described as a drawing (called a network) below:

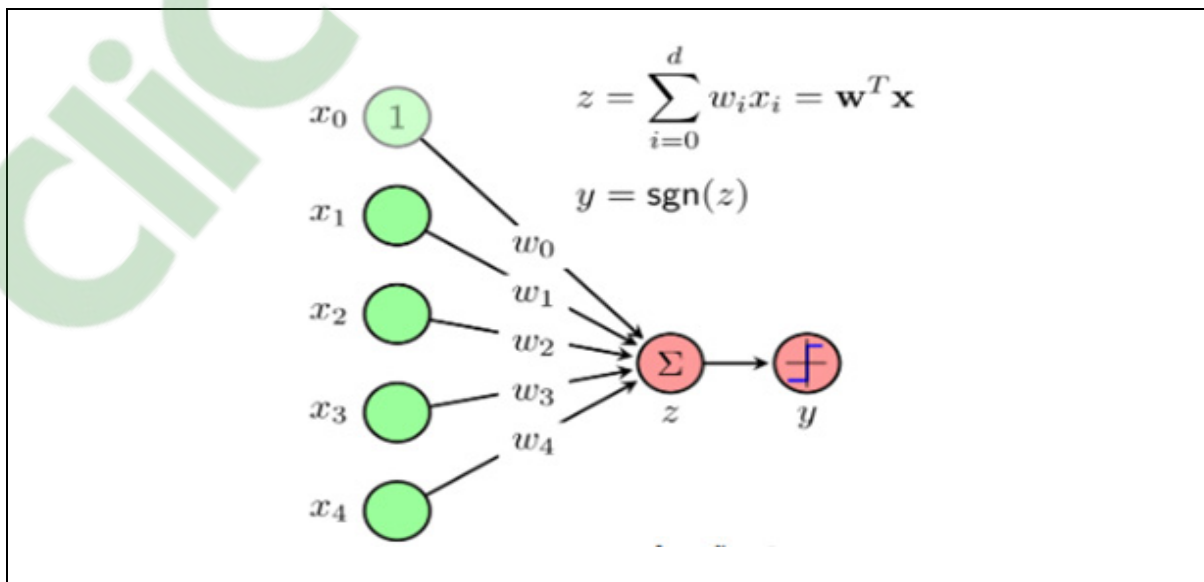


Figure 4.1 Perceptron Learning Algorithm
Taken from Vu Huu Tiep (2018, p.162)

The input of network x is illustrated by green nodes with the node always equal to 1. The set of green nodes is called the Input layer. In this example, we assume the dimension of data $d = 4$. The number of nodes in the input layer is always $d + 1$, with a node of 1 added. This Node $x_0 = 1$ is like a bias, b sometimes is hidden.

Weights x_0, x_1, \dots, x_d are assigned to arrows to node $z = \sum w_i x_i = \mathbf{w}^T \mathbf{x}_i$. Node $y = f(z) = \text{sgn}(z)$ is the output of the network. The blue inverted z symbol in node y shows the graph of function sgn .

In the PLA algorithm, we have to find the weights points placed in the input layer. That for each x_i in the set of training data points placed in the Input layer, the output of this network matches the corresponding y_i label.

Function $y = \text{sgn}(z)$ is also called an activation function. This function is the simplest form of neural network.[42]

Next-generation neural networks may have multiple nodes at the output forming an output layer or also may be additional intermediate layers between the input layer and the output layer. Those intermediate layers are called hidden layers. When performing large networks, people often simplify the picture in which node $x_0 = 1$ is usually hidden. The z node is also hidden and written into the y node. Notice that if we change the activation function by $y = z$, we will have a neural network describing the Linear Regression algorithm

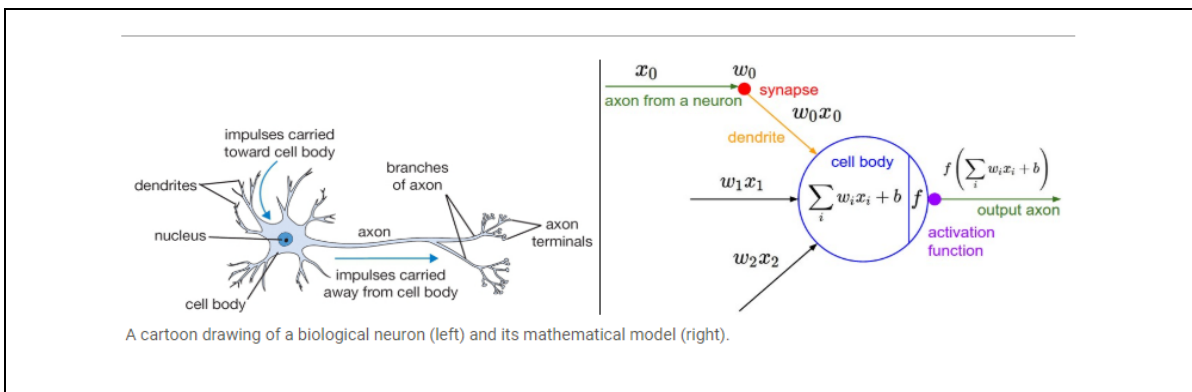


Figure 4.2 Neural Network describing the Linear Regression algorithm

Taken from Karpathy (2017)

www: <http://cs231n.github.io/neural-networks-1/>
(Retrieved from website on February 27, 2020)

A single-input neuron is shown in Figure 4-2. The scalar input p or x is multiplied by the scalar weight w to form $w^T x$, one of the terms that are sent to the summer. The other entries are multiplied by one way and then transferred to the summer. The summer output, often called the net input, enters an activation function, creating a scalar neuron output.

The neuron output is calculated $y = a = f(z) = f(w^T x + b)$. The function $a = f(z)$ is activation function or transfer function.

The transfer function can be a linear or nonlinear function of z ($a = \sigma(z)$). Most transfer functions are nonlinear functions except for the output function that can use linear functions. Each specific transfer function has been chosen to answer the issues that the neural network system is designed to solve. Three of the most commonly used activation functions are Sigmoid, Tanh, ReLU, which are discussed in detail below [35][43].

4.2 Activation function

The hard limit activation function is shown on the left of Figure 4-3, setting the neuron output to 0 if the value of the function is less than 0 or 1 if the value of the function is equal to or greater than 0 [43].

This function is often used to generate neurons that classify inputs within two separate classes.

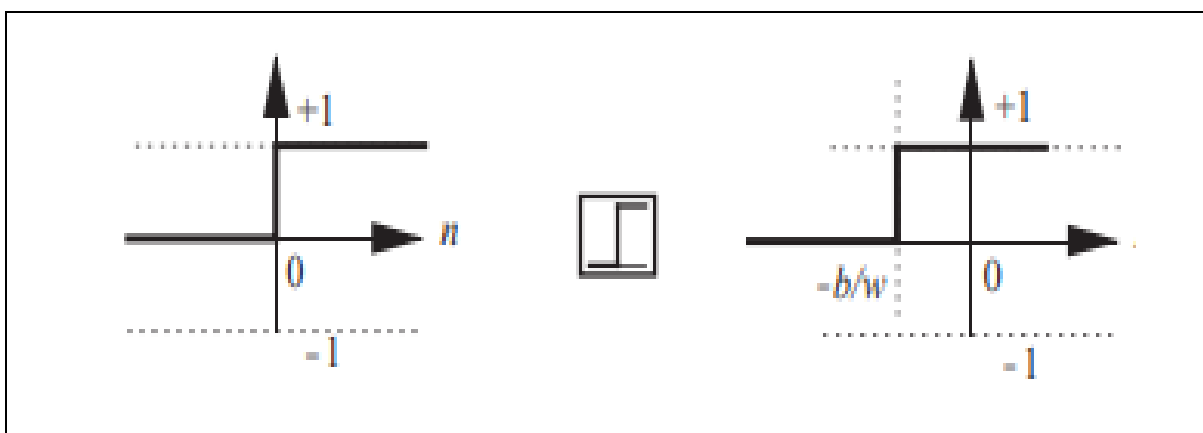


Figure 4.3 Hard Limit activation function

Taken from H. B. Demuth et al (2014)

In a linear activation function, the output is equal to its input ($a = z$). It is shown below [43]:

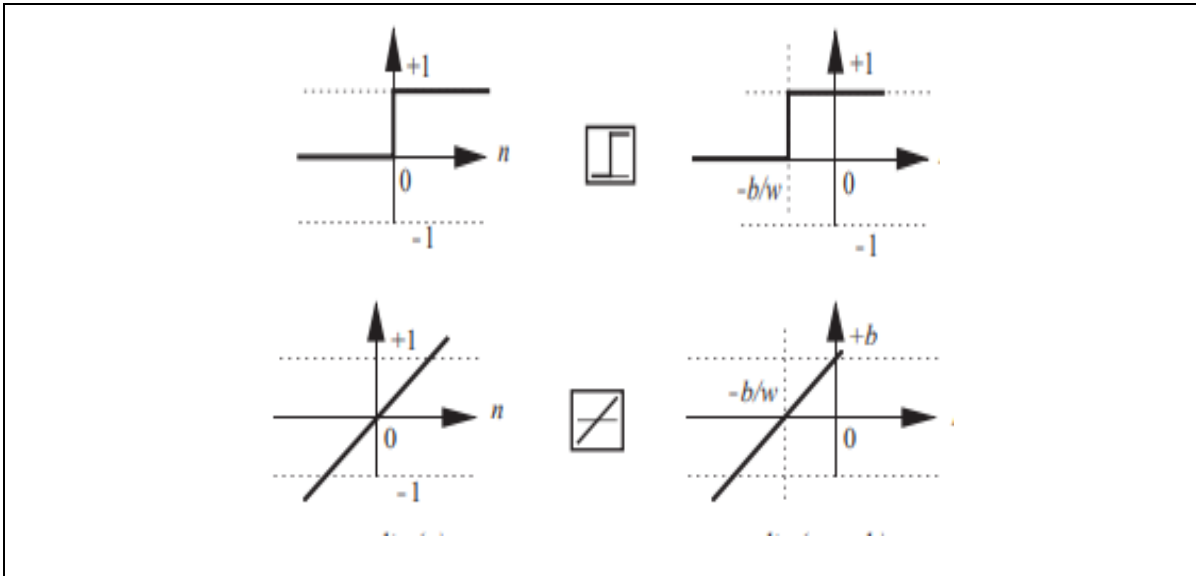


Figure 4.4 Linear activation function.
Taken from H. B. Demuth et al (2014)

4.2.1 Log-sigmoid activation function

The log-sigmoid activation function is shown below:

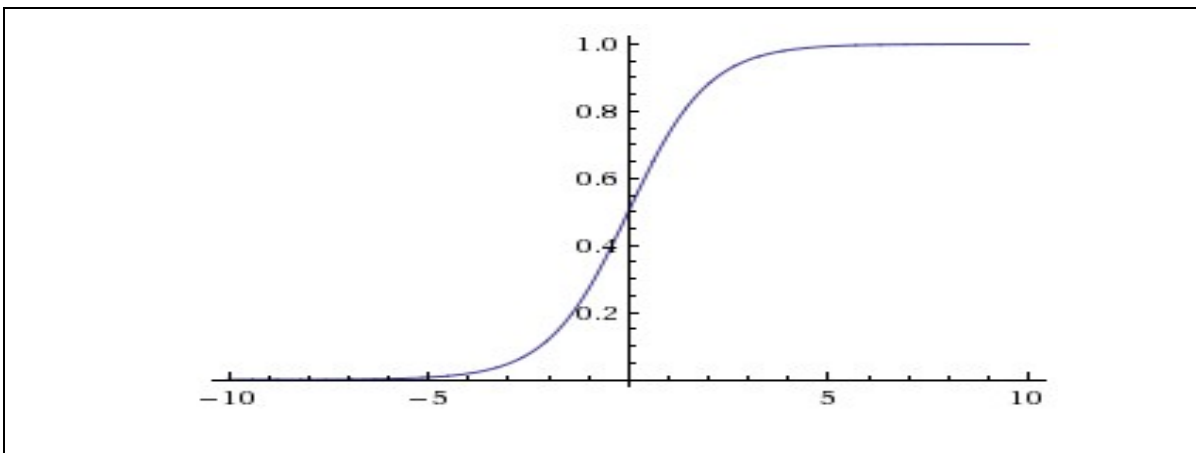


Figure 4.5 Sigmoid activation function
Taken from Karpathy (2017)
www: <http://cs231n.github.io/neural-networks-1/>
(Retrieved from website on February 27, 2020)

This activation function can take the input from negative infinity and positive infinity, and compresses the output in the scale of 0 to 1, in the formula below:

$$a = \sigma(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-wx}} \quad (4.1)$$

Sigmoid activation functions $a = \sigma(z)$ this activation function has a derivative a' :

$$\begin{aligned} a' &= \frac{e^{-z}}{(1 + e^{-z})^2} = \frac{1}{(1 + e^{-z})^2} \frac{e^{-z}}{(1 + e^{-z})^2} \\ &= \sigma(z) * (1 - \sigma(z)) \end{aligned} \quad (4.2)$$

4.2.2 Tanh activation function

Similar to the sigmoid activation function, we have the hyperbolic tangent activation function. This transfer function takes the input (there can be any value between negative infinity and infinity) and compresses the output in the -1 range at 1. The Tanh function is shown below:

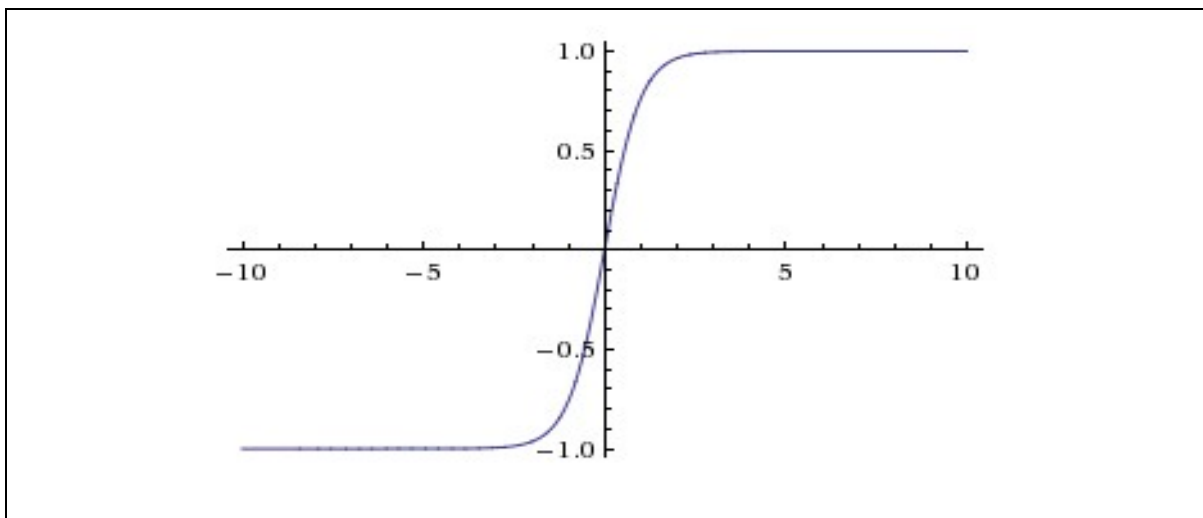


Figure 4.6 Tanh activation function
 Taken from Karpathy (2017)
 www: <http://cs231n.github.io/neural-networks-1/>
 (Retrieved from website on February 27, 2020)

$$a = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} = \frac{e^{Wx} - e^{-Wx}}{e^{Wx} + e^{-Wx}} \quad (4.3)$$

Tanh activation functions $a = \tanh(z)$ this activation function has a derivative a' :

$$a' = \frac{(e^{-z} + e^{-z})^2 - (e^{-z} - e^{-z})^2}{(e^{-z} + e^{-z})^2} = 1 - \tanh(z)^2 \quad (4.4)$$

Similar between Sigmoid and Tanh activation function because $\tanh(z) = 2\sigma(2z) - 1$.

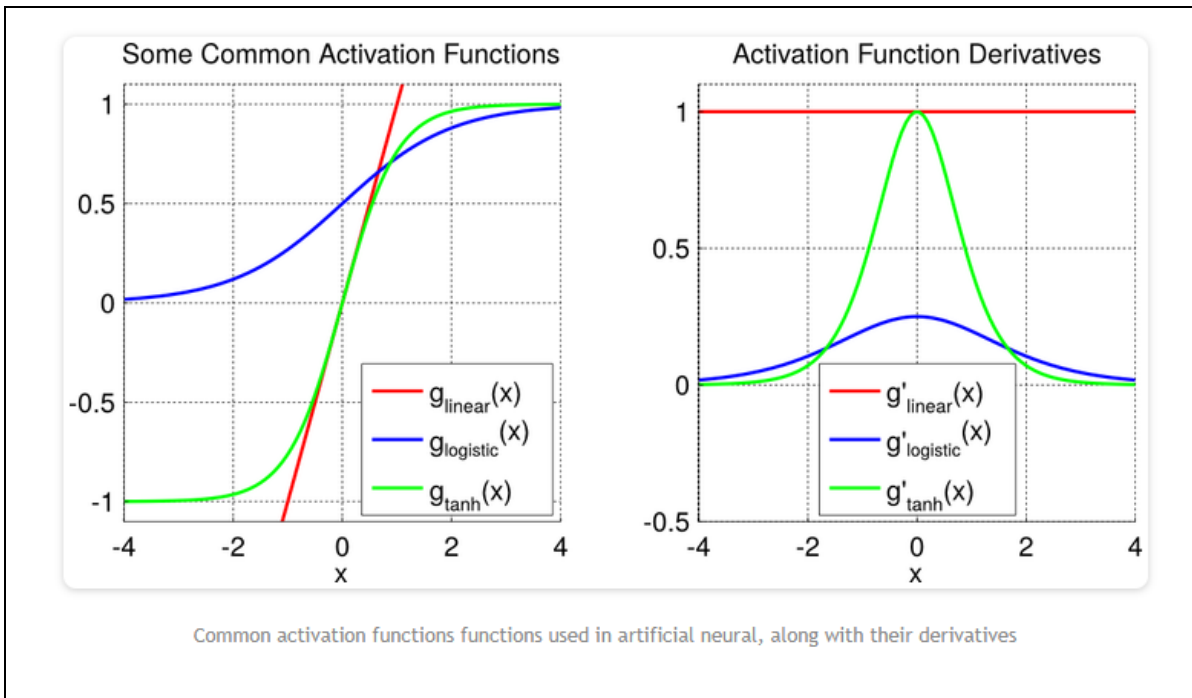


Figure 4.7 Linear, Sigmoid, and Tanh activation function, along with their gradient
Taken from Stansbury (2014)

www: <https://theclevermachine.wordpress.com/2014/09/08/derivation-derivatives-for-common-neural-network-activation-functions/>
(Retrieved from website on February 27, 2020)

Sigmoid activation function has two significant drawbacks:

- 1) Sigmoid saturate and kill gradients (max gradient sigmoid function = $1/4$).
- 2) Sigmoid outputs are not zero-centered. Consequently, zero-centered is trouble, but this trouble has less effect compared to the beginning saturated activation problem.

Similar to the sigmoid neuron, the Tanh activation function also saturates and kills gradient (max gradient sigmoid function = 1). Nevertheless, the Tanh activation function has an output that is zero-centered. Hence, the Tanh function is always preferred in the application compared to the sigmoid function.

The log-sigmoid activation function is often used to simulate the nonlinear problem in simple multilayer networks (one hidden layer) formed by a backpropagation algorithm. Log sigmoid activation function also famously applies for a problem relevant to probability and classification. [43]

4.2.3 ReLU activation function

ReLU (Rectified Linear Unit) has been widely used in deep learning networks recently because of its simplicity. The graph of the ReLU function is illustrated in Figure 4-7 (left). It has the math formula $f(s) = \max(0, s)$ - very simple. Its main advantages are:

- ReLU is proven to make training for Deep Networks much faster (Krizhevsky et al.). Figure 4-7 (right) compares the convergence of SGD when using two different activation functions: ReLU and Tanh. This acceleration is thought to be because ReLU is calculated almost instantaneously. ReLU's gradient is also calculated very fast with a gradient of 1 if the input is greater than 0, equal to 0 if the input is less than 0.
- Although ReLU has no derivative at $s = 0$, in practice, it is still common to define $\text{ReLU}'(0) = 0$ and further assert that the probability of the input of a zero unit is very small.

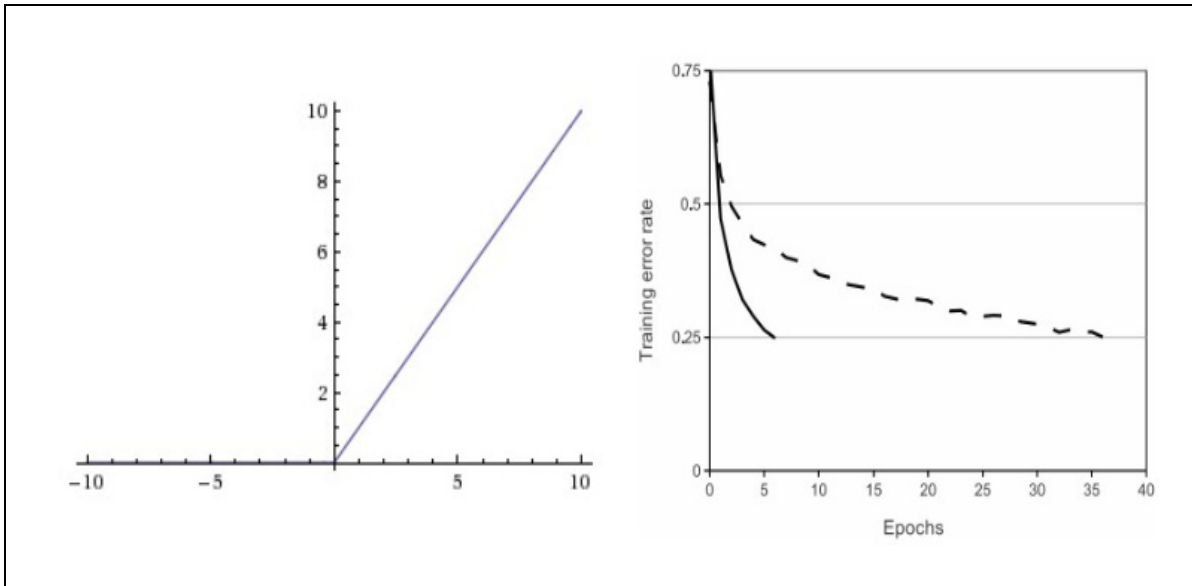


Figure 4.8 ReLU Rectified Linear Unit activation function

Taken from Karpathy (2017)

www: <http://cs231n.github.io/neural-networks-1/>

(Retrieved from website on February 27, 2020)

4.3 Multilayer neural network

Perceptron Learning Algorithm (PLA) was considered the representation of fundamental binary problems. Represent logical functions are NOT, AND, OR, and XOR (output equals one if and only if two different input). To be able to use PLA (output is 1 or -1), we will replace the values 0 of the output of these functions by -1. In the top row of Figure 4-9 below, the blue square points are the points with label equal to 1, and the red circle points are the points with label equal to -1. The lower row of Figure 4-9 is the perceptron model with the corresponding coefficients.

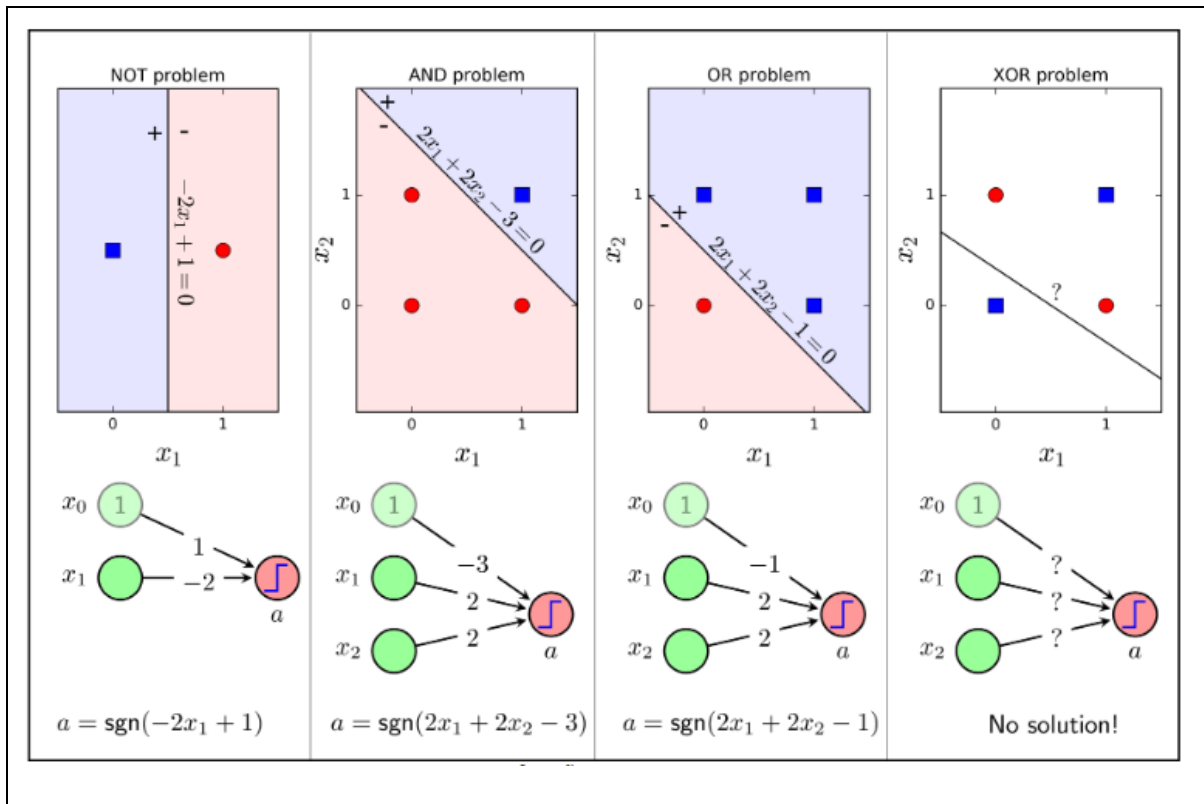


Figure 4.9 Perceptron Learning Algorithm (PLA) for fundamental binary problems
Taken from Vu Huu Tiep (2018, p. 194)

Notice that for the OR, AND, and OR problems, the data is linearly separable, so we can find the coefficients for the perceptron to represent each function correctly.

With the XOR function, because the data is not linearly separable, it is impossible to find a line that divides the red and green layers, so the problem is inexperienced. If we replace PLA with Logistic Regression, i.e., replacing the function activation function from sgn to sigmoid . We also cannot find the satisfied coefficients, because, Logistic Regression also only creates borders with linear form. Therefore, the Neural Network models we know cannot represent this simple logical function XOR.

Notice that if two straight lines are allowed, the problem of representing the XOR function will be solved, as shown in Figure 2 (left) below:

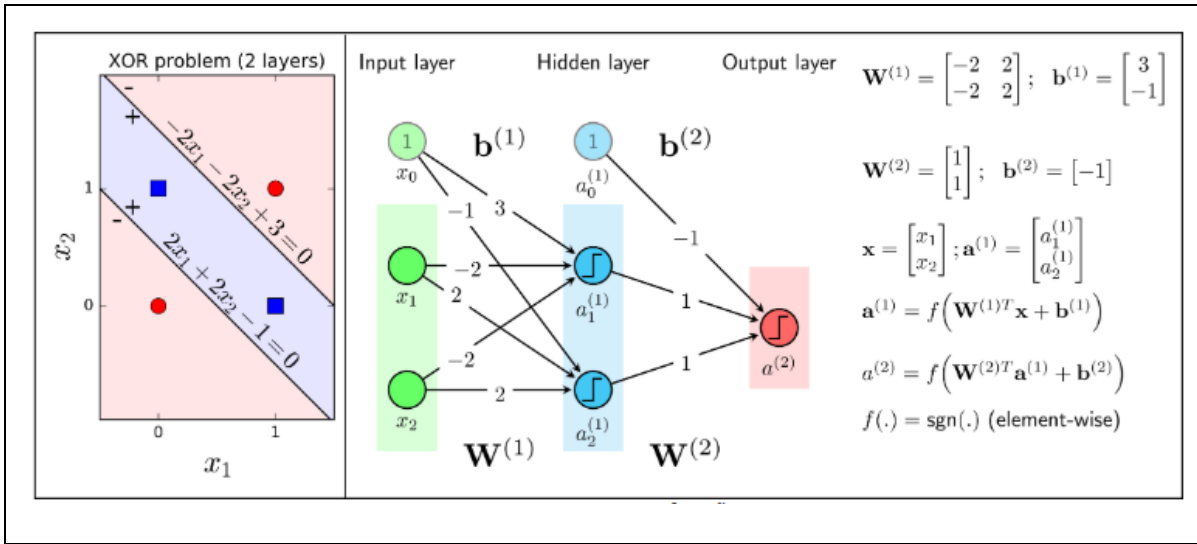


Figure 4.10 XOR function solved by a neural network with a hidden layer
 Taken from Vu Huu Tiep (2018, p. 194)

In 1989 the paper was shown that neural networks with at least one hidden layer are universal approximators [44]. According to this paper, this neural network with at least one hidden layer can approximate any continuous function. Michael Nielsen also was intuitive to explain that in his online book [45]. If given any continuous function $f(x)$ and some $\epsilon > 0$, there exists a Neural Network $g(x)$ with one hidden layer (with a reasonable non-linearity active function of, e.g., sigmoid) such that $\forall x, |f(x) - g(x)| < \epsilon$.

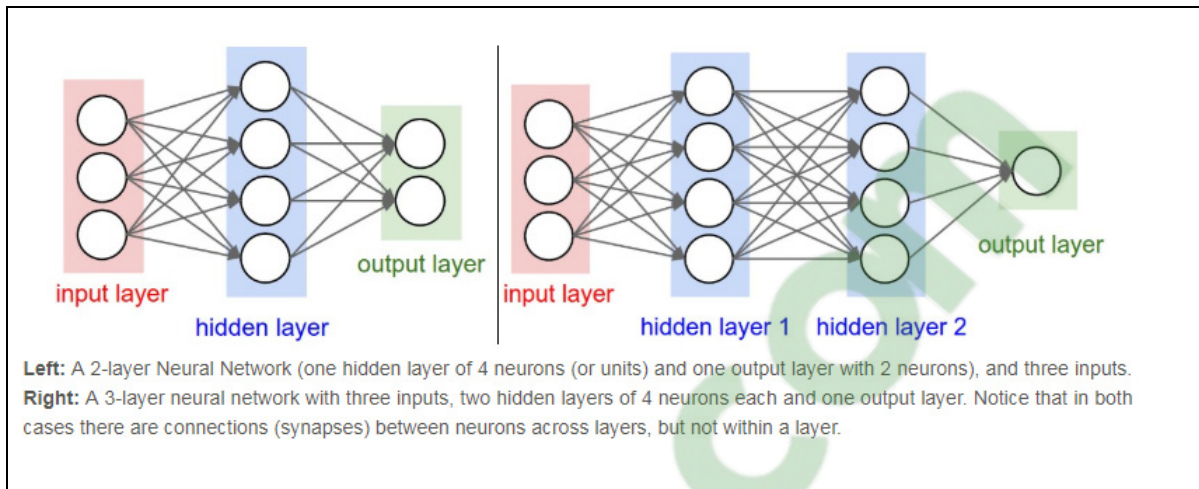


Figure 4.11 “Artificial Neural Networks”(ANN) or “Multi-Layer Perceptrons”(MLP)

Taken from Karpathy (2017)

www: <http://cs231n.github.io/neural-networks-1/>

(Retrieved from website on February 27, 2020)

Architecture neural network is a critical problem. The architecture of the network system relates to the overall structure. For example, how many layers, how many units are in each layer, and the way connected each unit.

One of the important metrics measures the size of neural networks that people usually use is the number of neurons or the number of parameters. In other words, it is the sizing of neural networks.

We can see in the figure 4-2 to count neurons in the neural network, (we not counting the inputs), the left network has $4 + 2 = 6$ neurons $[3 \times 4] + [4 \times 2] = 20$ weights parameter and $4 + 2 = 6$ biases parameter, for a total of 26 learnable parameters.

The right network has $4 + 4 + 1 = 9$ neurons, $[3 \times 4] + [4 \times 4] + [4 \times 1] = 12 + 16 + 4 = 32$ weights parameter and $4 + 4 + 1 = 9$ biases parameter, for a total of 41 learnable parameters.

A class with output is the neural network output called the output layer. The other classes are called hidden layers. When increasing the number and size of layers in a neural network, the capability of the neural network also will increase.

The design and formation of a neural network are like other machine learning models. Neural networks often use gradient-based optimizers to train, control cost functions at a minimum

value, to find the parameters w and b for each neural network. The design of cost functions will often be convex functions as the cost function is the least-squares function. Optimizing convex convergence starts from any initial parameter (theoretically in the application, it's powerful but can face numerical obstacles). The gradient descent method utilized to non-convex loss functions does not guarantee to find the optimal globe value. The values of the initial parameters are sensitive to the gradient descent method. For feed-forward neural networks, the initialized all weights regularly chose by little random values. The initialized bias b also can be selected a small positive value or zero.

The neural network model can be set up with more than one hidden layer. The number of hidden classes, the number of neurons in each class are an issue that needs research and testing. Often complex problems will require models with more hidden layers and a more significant number of neurons. If problems that are not too complicated, most neural networks currently have only two or three layers to meet the requirements.

Gradient descent learning can be done efficiently and accurately in the neural network. The simple machine learning algorithms explained in this chapter can be worked adequately on many significant problems. However, for more challenging problems such as speech recognition and image recognition, simple neural algorithms and models cannot be solved with high accuracy. In recent years, the development of the deep learning model has had specific success in addressing these challenging issues. First, the challenge occurs for new examples. It is exponentially extra challenging when running by massive data. The second challenge is how to find the mechanisms used to achieve generalization, which in traditional machine learning are not enough to learn complex functions in a multidimensional space. This season also usually requires high computational costs. The deep learning model is created to overcome those barriers.

The Convolutional Neural Networks (CNN) model is an example of a deep learning model, which is a leading identification system, in which deep layers are an important component (for example, the order of 10 hidden layers). For this research, one argument is that the image holds a hierarchical formation (for example, faces recognized from the eyes, face edges to

form faces, etc.), so a lot of layers of treatment the thought has a visual meaning for this data domain [46].

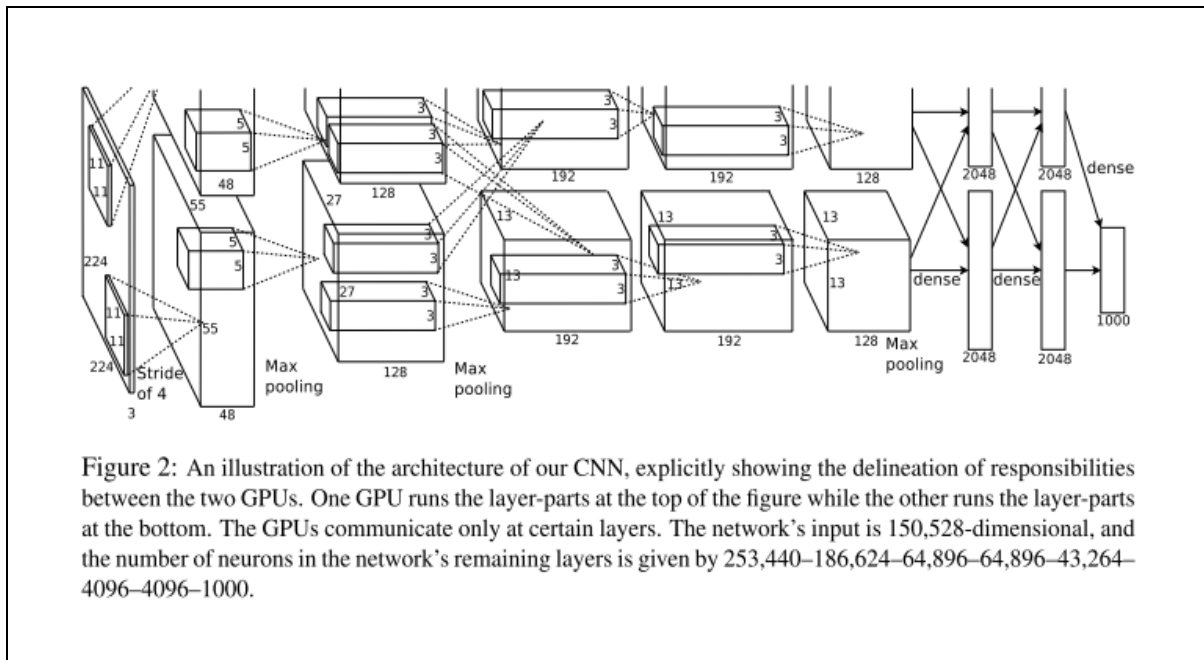


Figure 4.12 Deep Convolutional Neural Networks –AlexNet
Taken from A. Krizhevsky et al (2012)

4.4 Gradient base Method

The gradient is one of the popular methods to solve problems in a neural network as well as convex optimization problems. Machine learning has been designed to avoid general optimization by designing loss functions and constraints to ensure that optimization is convex. Because if loss function is non-convex functions, then solving the optimization problem, in general, is a difficult task [35].

For example, the Loss function $L(w) = \frac{1}{2N} \sum_{i=0}^N (w_i x_i - y_i)^2$ is convex.

Optimization algorithms in machine learning were used first-order and second order optimization [47].

Come back with an example in a recipe machine learning. One step is to calculate the gradient of the Loss function for each parameter.

A Recipe for machine learning

1. Given training data

$$\{x_i, y_i\}_{i=1}^N$$

2. Choose each of these:

Decision function

$$\hat{y} = f_{\theta}(x_i)$$

Loss function:

$$L(\hat{y}, y_i) \in \mathbb{R}$$

3. Define the goal:

$$\theta^* = \arg \min_{\theta} \sum_{i=1}^N L(f(x_i), y_i)$$

4. Train with stochastic gradient descent (take small steps opposite the gradient):

$$\theta^{(t+1)} = \theta^t - n_t \nabla L(f(x_i), y_i) \Rightarrow y_{new} = f_{\theta^*}(x_{new})$$

Figure 4.13 A recipe for machine learning

Taken from Gormley (2018)

www: <http://www.cs.cmu.edu/~mgormley/courses/10601-s18/slides/lecture11-nn.pdf>

(Retrieved from website on February 27, 2020)

The result of the neural network training process will find the values of w and b of the network. This process is also the need to find the minimum value of the loss function. One of the typical methods used in machine learning is gradient descent [31].

There is no guarantee that optimization algorithms will reach the local minimum in a short time. However, this is usually taking the minimum value of the loss function fast acceptable. Batch gradient descent will compute the gradient using the whole training dataset, while stochastic gradient descent (SGD) compute the gradient only using a single sample dataset.

The original stochastic gradient has many vital applications. The stochastic gradient is the primary way to form models on massive data.

Gradient descent is one of the most well-known algorithms for achieving optimization. Each modern Deep Learning library has many different algorithms to optimize gradient descent.

However, neural network algorithms are often used as black-box optimization because difficult find reasonable explanations of their strengths and their weaknesses [32].

By updating the parameters in the reverse direction of the gradient of the objective function $\nabla J(\theta)$ with respect to the parameters, we can find to minimize an objective function $J(\theta)$. That is the gradient descent method. We can imagine that we follow the direction of the slope of the surface created by the objective function downhill until we reach a valley. The learning rate η determines the size of the steps we take to reach a (local) minimum.

Gradient descent is a method to find the minimum of the objective function $J(\theta)$ (or the loss function). The objective function is expressed through model parameters. Model's parameters $\theta \in \mathbb{R}^d$ have updated the parameters in the opposite direction of the gradient of the objective function $\nabla J(\theta)$ with respect to the parameters. The parameters thought the training process by updating the model parameters inversely to the slope of the objective function. The size of the steps we take to reach a (local) minimum depends on the learning rate. With a significant learning rate, the update step will be massive, but it may be exceeded the minimum value of the objective function. A small percentage will update slowly. The training process will take longer.

In case neural network objective function $\nabla J(\theta)$ is Loss (Cost) function (noted that $\nabla J(w, b)$) and parameters are w, b ($b = w_0$). We can rewrite minimize an objective function $L(w)$ parameterized by a model's parameters $\theta \in \mathbb{R}^d$ by updating the parameters in the opposite direction of the gradient of the objective function $\nabla L(w)$ with respect to the parameters w [35][48].

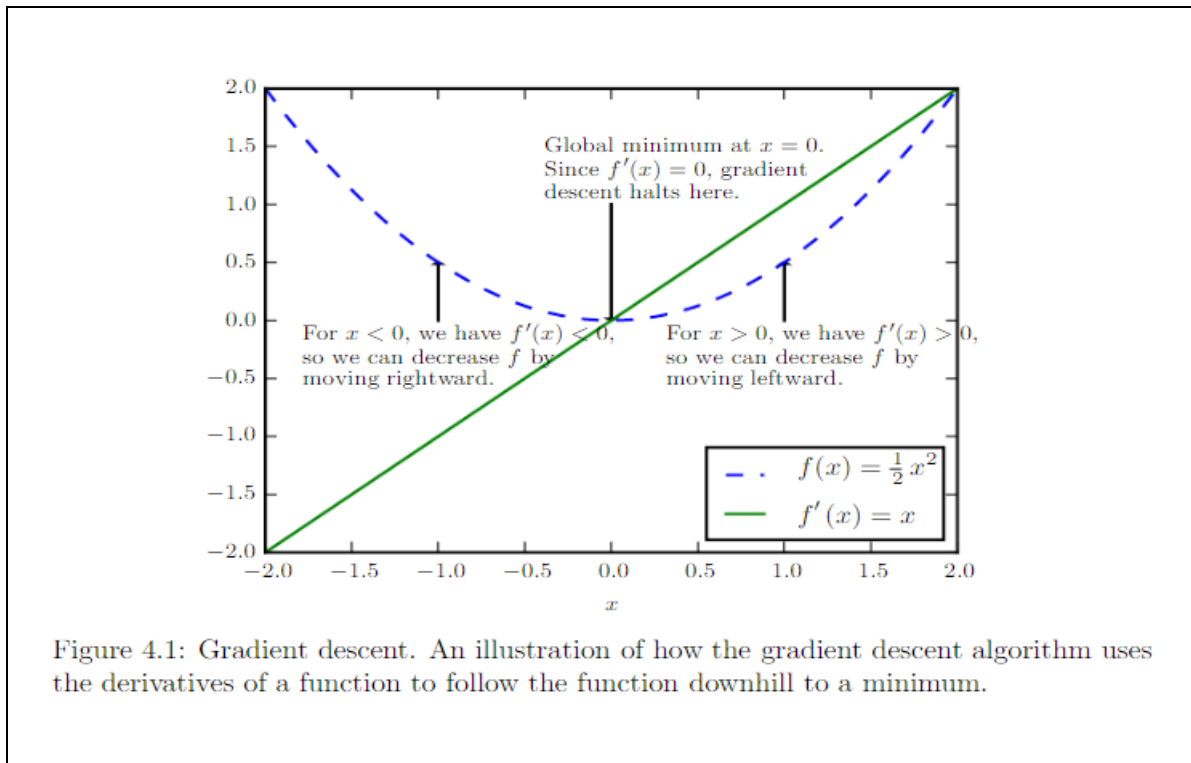


Figure 4.14 Gradient Descent Method
Taken from I. Goodfellow et al (2016, p.81)

4.4.1 Gradient descent variants

Batch, mini-batch, and stochastics are three variants of the gradient descent method. Depending on the amount of dataset, we will choose which variant of the descending derivative method. There will be a trade-off between updating the parameters accurately and the less accurate updates, but training time is faster to find the minimum value.

4.4.1.1 Batch gradient descent

Vanilla gradient descent or batch gradient descent computes the gradient of the cost function for all datasets before update parameters.

$$w_{t+1} = w_t - \eta \nabla L(w_t) \quad (4.5)$$

With η is the learning rate.

In our example, in the regression problem, the best result is when the difference between the predicted output and the output is the least square errors. The loss function of the regression problem shown below:

$$L(w) = MSE = \frac{1}{2N} \sum_{i=1}^N (y_i - w_i x_i)^2 \quad (4.6)$$

The gradient of the loss function shown below:

$$\nabla_w L(w) = MSE = \frac{1}{N} \sum_{i=1}^N (y_i - w_i x_i) x_i \quad (4.7)$$

For example, if we have dataset $N = 100\,000$ samples, then when we calculate follow batch gradient descent method. We need 100 000 times to calculate the average value $\nabla_w L(w)$

$$w_{t+1} = w_t - \eta \nabla L(w_t) = w_t - \eta \frac{1}{100000} \sum_{i=1}^{100000} (y_i - w_i x_i) x_i \quad (4.8)$$

Since we need to calculate the slope for the entire data set to perform a single update, the reduction of the slope of the batch can be prolonged and inaccessible to data sets. That does not fit into the memory. Mass amount computes also does not allow us to update the model online. Batch gradient descent is warranted to meet a local minimum for non-convex loss function and the global minimum for convex loss.

When we update the parameters in the reverse path of the slope, the learning rate (η) determining the large size of the update that we operate.

4.4.1.2 Stochastic gradient descent

Stochastic gradient descent is a branch of the gradient descent algorithm introduced in sections 4.4 and 4.5. A problem in machine learning is that it takes large training packages to generalize well. However, operating more massive training sets are also more expensive in computer terms.

A machine learning algorithm often decomposes a cost function as a sum over training examples of some per-example loss function.

Stochastic gradient descent (SGD) however, performs a parameter update for each training example x_i and label y_i , each iteration predicted this gradient based on a single randomly picked example:

$$w_{t+1} = w_t - \eta \nabla L(w_t, x_i, y_i) \quad (4.9)$$

Almost all deep learning library is provided by a fundamental algorithm: stochastic gradient descent (SGD). Batch gradient descent performs backup calculations for large data sets, as it recalculates the slope for similar examples before each parameter update. SGD eliminates this redundancy by performing an update at a time. Therefore, often much faster and can be used for online learning. SGD performs regular updates with high variance, resulting in a significant fluctuation of the target function, as shown in Figure 4-15 below:

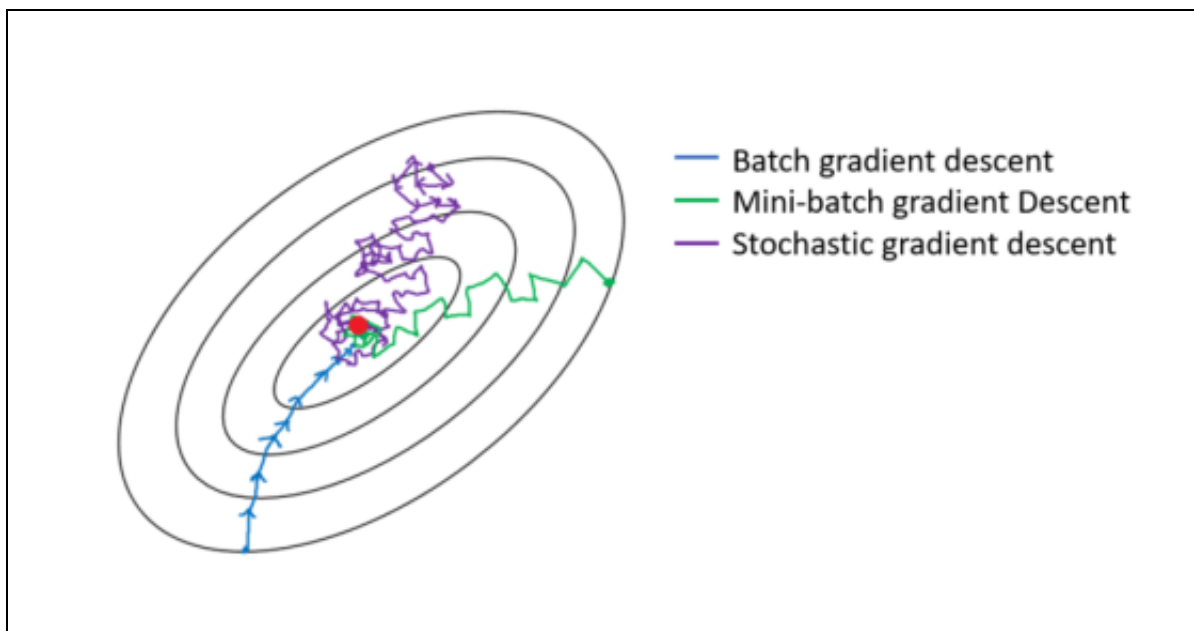


Figure 4.15 Performance of three gradient descent variant
Taken from Andrew Ng (2017)

As the downward hill in the movement finds to the minimum of the valley, the parameters are stored.

On the one hand, the movement of SGD allows it to switch to a new local minimum and is better suitable. On the other hand, this complicates to find the correct minimum, as SGD will continue to be fluctuation. However, it has been proven that as we slowly reduce the learning rate, SGD presents the same convergence behavior to find the minimum. Almost certainly converging to a local minimum or global minimum, with non-convex and convex problems, respectively.

4.4.1.3 Mini-batch gradient descent

Finally, the reduction of small-scale slopes will give the best results from both worlds and will provide updates for each example of small neural network training as below:

$$w_{t+1} = w_t - \eta \nabla L(w_t, x_{i:i+n}, y_{i:i+n}) \quad (4.10)$$

If in our example, $N = 100\,000$ and $n = 1000$ ($k = 1 + n - 1 = 100$ examples) then volume calculate gradient Loss function reduce 100 times.

$$w_{t+1} = w_t - \eta \nabla L(w_t) = w_t - \eta \frac{1}{1000} \sum_{i=1}^{1000} (y_i - w_i x_i) x_i \quad (4.11)$$

Like the stochastic gradient descent, Mini-batch gradient descent also based on randomly picked examples for each iteration to calculate this gradient. One point to note is that, after each epoch, we need to shuffle the order of the data to ensure randomness.

We have two ways that are below:

It is a) reduce the variance of parameter updates, which can lead to more stable convergence, and; b) it is possible to use highly optimized matrix optimizations for modern deep learning libraries to facilitate the computation of the with respect to a mini-batch is beneficial.

The sizes of typical small batches range from 32 to 256 and are variable for different applications. The decreasing slope of small batches is often the algorithm chosen when forming neural networks, and the term SGD is often used when using small batches.

Minimizing a loss function is often equivalent to maximum likelihood [35].

4.5 First-order and second-order optimization algorithms

4.5.1 First-order optimization algorithms

There are three primary ways of how these optimizers can affect the gradient descent:

- (1) Changing the learning rate component, η , or
- (2) Changing the gradient component, $\partial L/\partial w$, or
- (3) Changing both.

$$w_{t+1} = w_t - \eta \frac{\partial L}{\partial w} \quad (4.12)$$

Notations:

t: time step

w: weight/parameter, which we want to update

η : learning rate

$\partial L/\partial w = \nabla J(\theta)$: gradient of L, the loss function to minimize, with respect to θ .

For the first way, these optimizers times by a positive factor to the learning rate, such that they become smaller (e.g., RMS prop).

For the second way, instead of just taking one value like in vanilla gradient descent, optimizers will apply the moving averages of the gradient (momentum).

Adam and AMS Grad are optimizers that can act in both two ways [48].

Table 4.1 Three approach of the variants first-order gradient optimizers
 Take from Karim (2018)
 www: <https://towardsdatascience.com/10-gradient-descent-optimisation-algorithms-86989510b5e9>
 (Retrieved from website on February 27, 2020)

Optimiser	Year	Learning Rate	Gradient
Momentum	1964		Yes
AdaGrad	2011	Yes	
RMSprop	2012	Yes	
Adadelta	2012	Yes	
Nesterov	2013		Yes
Adam	2014	Yes	Yes
AdaMax	2015	Yes	Yes
Nadam	2015	Yes	Yes
AMSGrad	2018	Yes	Yes

Figure 4.16 below shows how these optimizations transformed and developed from a pure vanilla random gradient (SGD), modification to Adam's variants. SGD is divided into two main types of optimization. The first is the types of optimization, improving through the learning rate component, through the momentum component. The second is the gradient component.

Momentum and Nesterov will be explained below, and the other detail formula was shown in [48].

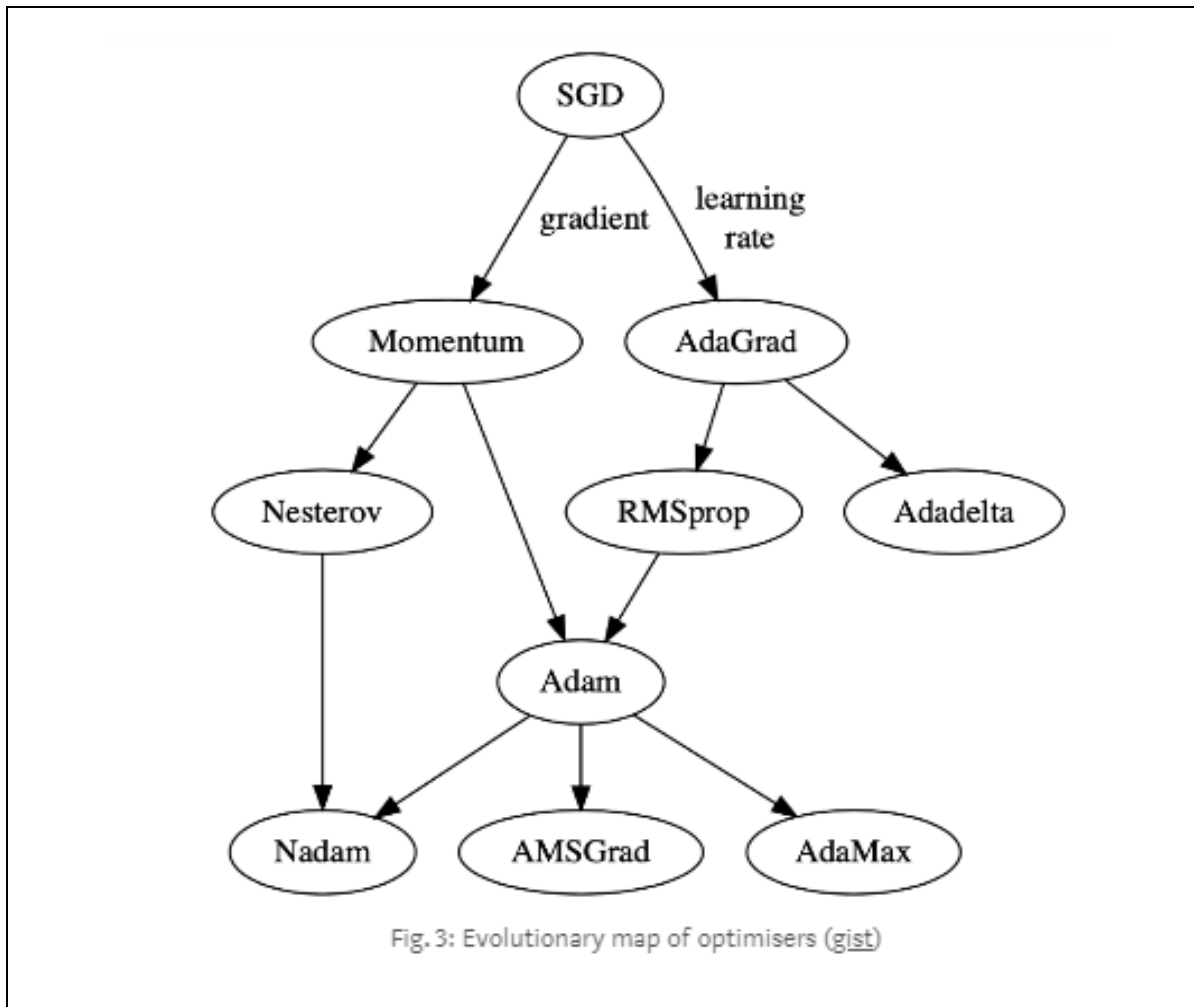


Figure 4.16 Develop and transform of the first-order gradient optimizers

Take from Karim (2018)

www: <https://towardsdatascience.com/10-gradient-descent-optimisation-algorithms-86989510b5e9>

(Retrieved from website on February 27, 2020)

4.5.1.1 Momentum

Stochastic gradient descent (SGD) has difficulties in navigating valleys when they are navigating through curved surfaces in one dimension, which is common around local optima. However, in image c) gradient descent with momentum, the SGD oscillates faster and across the slopes of the valley along the bottom towards the local optimum with momentum.

It adds momentum volume by adding a fraction γ of the new vector of the previous step to the current update vector:

Gradient Descent or Vanilla Gradient Descent was shown below:

$$w_{t+1} = w_t - \eta \frac{\partial L}{\partial w} \quad (4.13)$$

Momentum Gradient Descent was shown below:

$$v_{t+1} = \gamma v_{t+1} + \eta \frac{\partial L}{\partial w} \quad (4.14)$$

$$w_{t+1} = w_t - \gamma v_{t+1} \quad (4.15)$$

$$w_{t+1} = w_t - \gamma v_{t+1} - \eta \frac{\partial L}{\partial w} \quad (4.16)$$

Note: The value γ is normally set to 0.9. Some implementations switch the signs in the equations.

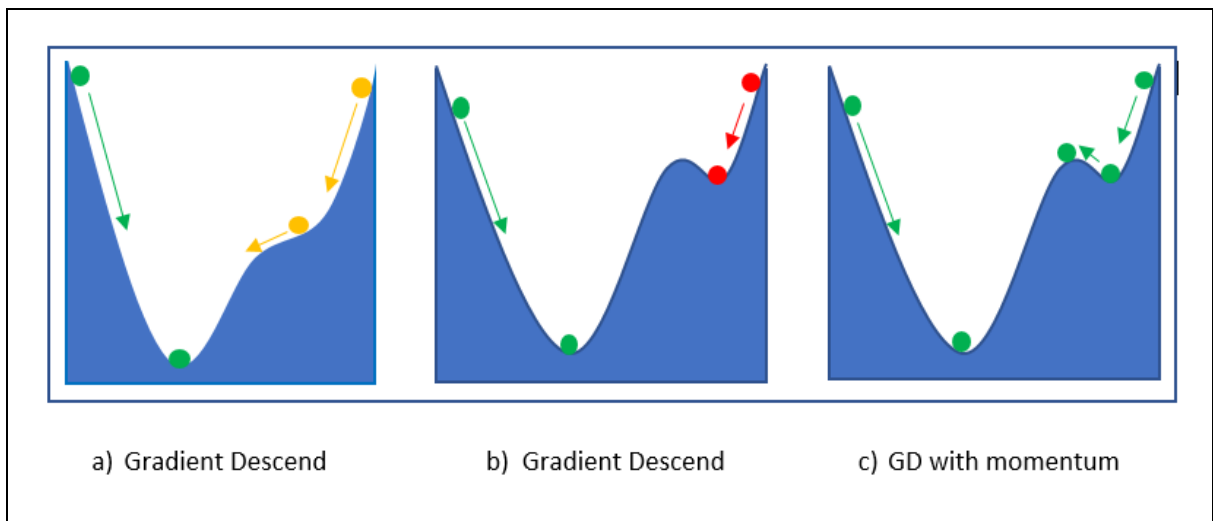


Figure 4.17 Compare Gradient Descent with physical phenomena
Taken from Vu Huu Tiep (2018, p.149)

For example, in Figure 4.17, when using momentum, we kick a ball down a hill. As it rolls downhill, the ball accumulates momentum and starts to get faster and faster on its way. Until

the ball reaches its final velocity with the fact that there is air resistance, $\gamma < 1$). The same thing applies to our parameter updates. For dimensions whose gradients point in the same directions, the momentum term increases, however, for those that change directions, the momentum reduces updates for dimensions. Finally, we acquire faster convergence and decreased oscillation.[42]

4.5.1.2 Nesterov accelerated gradient

To change our momentum term this kind of prediction, we can use a new way called Nesterov accelerated gradient (NAG).

With standard momentum: the amount of change is the sum of two vectors: momentum vector and gradient at present.

With Nesterov momentum: the amount of change is the sum of two vectors: the vector momentum and the gradient at the time is approximately the next point.

$$v_t = \gamma v_{t-1} + \eta \frac{\partial L}{\partial (w_t - \gamma v_{t-1})} \quad (4.17)$$

$$w_{t+1} = w_t - v_t \quad (4.18)$$

$$w_{t+1} = w_t - \gamma v_{t-1} - \eta \frac{\partial L}{\partial (w_t - \gamma v_{t-1})} \quad (4.19)$$

Again, the value of momentum term γ to was set approximately 0.9. During when the momentum first computes the continuous gradient (the small blue vector in Figure 4-15) and then takes a massive jump in the direction of the updated accumulated gradient (significant blue vector). NAG first increases up towards the previously collected gradient (the brown vector). It then measures the gradient and makes a correction (the red vector), which created in the complete NAG update (the green vector). This anticipatory update stops us from going too fast. As a result, it will increase responsiveness, which has a significant impact on the increase of the performance of Recurrent Neural Networks (deep network) on many tasks.

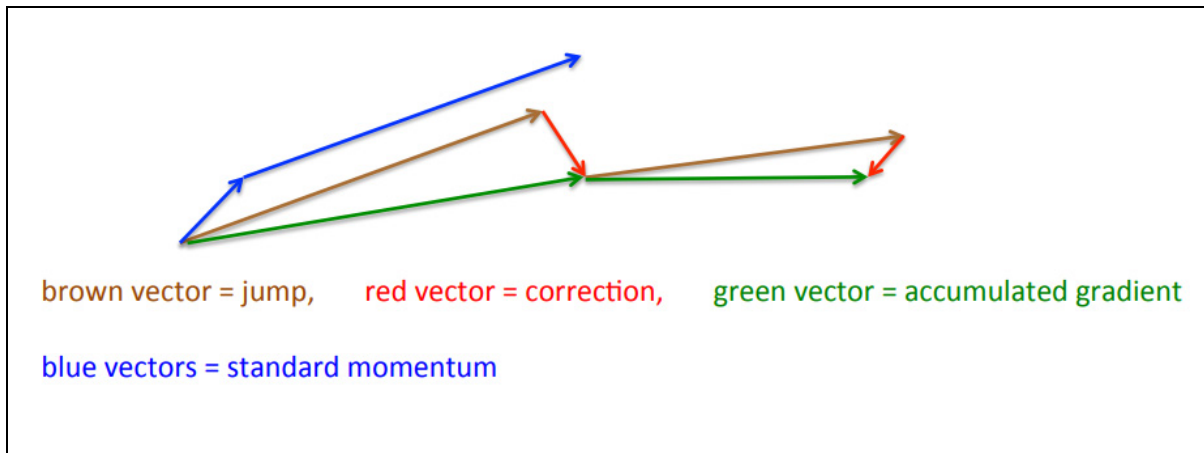


Figure 4.18 Nesterov update
 Take from Geoffrey Hinton et al
[www:https://www.cs.toronto.edu/~hinton/coursera/lecture6/lec6.pdf](https://www.cs.toronto.edu/~hinton/coursera/lecture6/lec6.pdf)
 (Retrieved from website on February 27, 2020)

This image can be an explanation of the intuitions behind the Nesterov accelerated gradient, while Ilya Sutskever shows more in detail with his Ph.D. thesis.

4.5.2 Second-order optimization algorithms

Newton's method is a second-order method; the way the solution requires calculating the second derivative. There is an algorithm that helps solve the problem $g(x) = 0$. We can solve the first-order gradient of the Loss function equal to zero by apply the Newton method.

For a variable function, the goal is to find the value of x to satisfy the equation $g(x) = 0$. Newton's method comes from approximating the value of the derivative at a point:

$$g'(x_1) = \frac{dg(x_1)}{dx} = \frac{g(x_1) - g(x_2)}{x_1 - x_2} \quad (4.20)$$

$$x_1 - x_2 = \frac{g(x_1) - g(x_2)}{g'(x_1)} \quad (4.21)$$

$$x_2 = x_1 - \frac{g(x_1)}{g'(x_1)} + \frac{g(x_2)}{g'(x_1)} \quad (4.22)$$

Since we are looking for solutions x_2 such that $g(x_2) = 0$, so with $g(x_1) = 0$, the above equation (4.22) becomes:

$$x_2 = x_1 - \frac{g(x_1)}{g'(x_1)} \quad (4.23)$$

In general, Newton's method of repeatedly repeating to find solutions is closer to the real value, with the experimental formula being with some initial value x_0 .

With the multi-variables function, we have J - Jacobian first-order gradient and H - Hessian second-order gradient.

$$w_{t+1} = w_t - \frac{\frac{\partial L}{\partial w}}{\frac{\partial^2 L}{\partial w^2}} \quad (4.24)$$

$$J(w) = \frac{\partial L}{\partial w} \quad (4.25)$$

$$H(w) = \frac{\partial^2 L}{\partial w^2} = J^T(w)J(w) \quad (4.26)$$

$$w_{t+1} = w_t - H^{-1}(w)J(w) \quad (4.27)$$

Similar to the quasi-Newton methods, we designed the Levenberg-Marquardt algorithm to proceed toward the second-order training speed without the necessity to compute the Hessian matrix. Once the performance function has the form of a sum of squares (as is typical in training feedforward networks), then the Hessian matrix can be approximated as:

$$H = J^T J \quad (4.28)$$

In addition, the gradient can be computed as

$$g = J^T e \quad (4.29)$$

Where J is known as the Jacobian matrix that has first derivatives of the network errors regarding the weights and biases, and e is known as a vector of network errors. We can

compute the Jacobian matrix through a standard backpropagation technique [49] that is much less complex than computing the Hessian matrix.

The Levenberg-Marquardt algorithm uses this approximation to the Hessian matrix in the following Newton-like update:

$$w_{t+1} = w_t - [J^T J + \mu I]^{-1} J^T e \quad (4.30)$$

Matrix I is the identity matrix. In the case of the scalar μ being zero, this is just Newton's method, with the use of the approximate Hessian matrix. If μ is large, this turns into a gradient descent with a small step size. At each iteration of the algorithm, the performance function is always decreased in this way.

The article [50] shows the original description of the Levenberg-Marquardt algorithm. The article [49] and the beginning of page 12-19 of the book [43] described the application of Levenberg-Marquardt to neural network training. The article [51] This algorithm seems to be the most rapid method for training moderate-sized feedforward neural networks (up to hundreds of weights). It also can be implemented efficiently in MATLAB® software. Since the solution of the matrix equation is a built-in function, so its attributes become even more pronounced in a MATLAB environment.

Based on the Levenberg-Marquardt optimization method, Bayesian regularization is a network training function that updates the weight and bias values. It minimizes a combination of weights and squared error, then determines the correct combination to produce a network that generalizes well [43].

Table 4.2 Several second-order optimization methods in MATLAB

Adapted from Mathworks.com

[www:https://www.mathworks.com/help/deeplearning/ug/choose-a-multilayer-neural-network-training-function.html;jsessionid=da9f87ea32f3896dd20868df6273](https://www.mathworks.com/help/deeplearning/ug/choose-a-multilayer-neural-network-training-function.html;jsessionid=da9f87ea32f3896dd20868df6273)

Acronym	Algorithm	Description
LM	<u>trainlm</u>	Levenberg-Marquardt
BR	<u>trainbr</u>	Bayesian regularization
BFG	<u>trainbfg</u>	BFGS Quasi-Newton
SCG	<u>trainscg</u>	Scaled Conjugate Gradient
CGB	<u>traincgb</u>	Conjugate Gradient with Powell / Beale Restarts
CGF	<u>traincgf</u>	Fletcher-Powell Conjugate Gradient
CGP	<u>traincgp</u>	Polak-Ribière Conjugate Gradient

4.6 Backpropagation

In the previous section, we saw how neural networks could learn their weights and biases by update by using the gradient descent algorithm. However, there remained a gap in our explanation of how to compute the gradient of the cost function. Therefore, in this section, an algorithm is known as backpropagation, a fast algorithm for computing such gradients will be presented.

In the 1970s, the backpropagation algorithm was initially introduced. However, its effect was not fully appreciated until a well-known paper by David Rumelhart, Geoffrey Hinton, and Ronald Williams in 1986 [52]. That paper illustrates several neural networks where backpropagation works considerably faster than earlier approaches to learning. The backpropagation was making neural nets possible to solve the problems. The

backpropagation was making neural nets possible to solve the problems. The problem had not been solvable before. Today, the backpropagation algorithm is an essential algorithm of learning in neural networks [45].

4.6.1 Concept backpropagation

Backpropagation is interesting content and has a significant influence on the calculation of parameters in the neural network method. We need to calculate the derivative base on the derivative chain rule to update the parameters. The forward and backward calculations are applicable, but with a neural network that has many parameters and input parameters, the reverse calculation can bring very significant results.

Forward-mode differentiation tracks how one input affects every node. Forward-mode differentiation begins at an input to the graph and moves forward to the end. It adds all the pathways connecting at every node. Each of those pathways describes the individual access in which the input influences that node. We get the whole channels in which the input influences the node by sums them up, and it is derivative. We can see that forward-mode differentiation is a simple chain rule technique calculus.

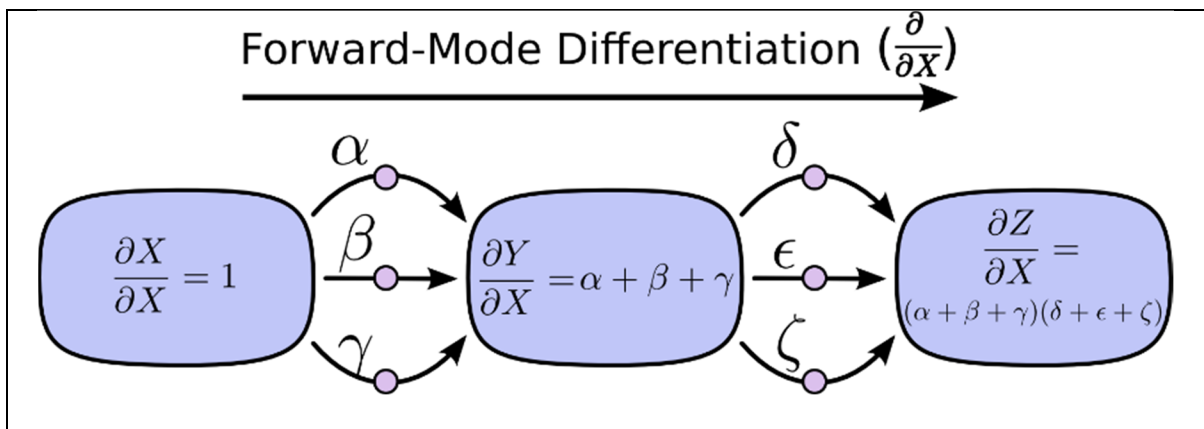


Figure 4.19 Forward-Mode differentiation

Taken from Olah (2015)

www: <https://colah.github.io/posts/2015-08-Backprop/>
(Retrieved from website on February 27, 2020)

By contrast, Reverse-mode differentiation tracks how each node affects one output. That is, forward-mode differentiation applies the operator $\partial/\partial X$ to all nodes, while reverse mode differentiation applies the operator $\partial Z/\partial$ to all nodes[53].

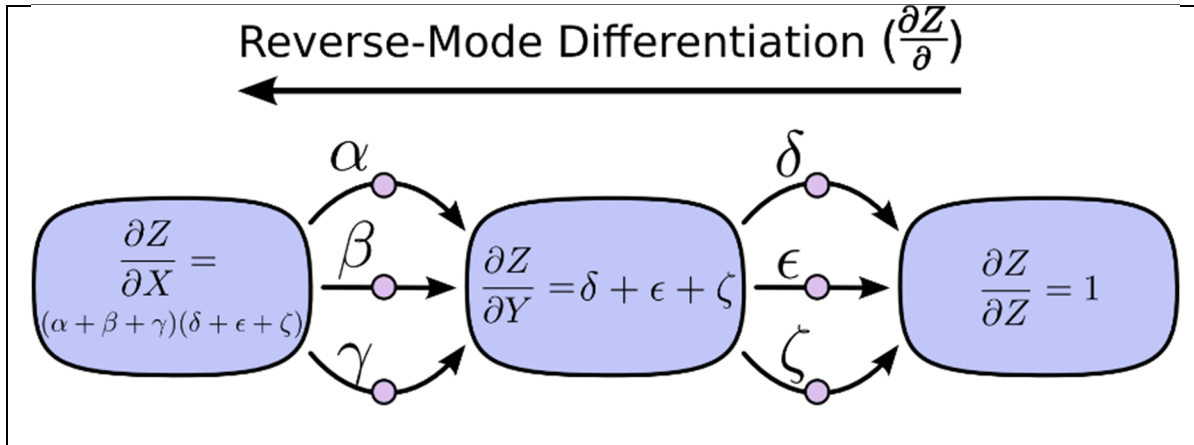


Figure 4.20 Reverse-Mode differentiation

Taken from Olah (2015)

www: <https://colah.github.io/posts/2015-08-Backprop/>
(Retrieved from website on February 27, 2020)

4.6.2 Apply backpropagation in neural network

The most common method to optimize multilayer neural networks or MLP (Multi-Layer Perceptrons) is a Gradient method was represented 4.4 and 4.5 in this thesis.

For example, to apply the Gradient Descend method. We need to calculate the derivatives of the Loss function $L(\mathbf{W}, \mathbf{b}, \mathbf{X}, \mathbf{Y})$ with respect to all the weight matrix $\mathbf{W}(l)$ and the bias vector $\mathbf{b}(l)$ of each neural. The neural was connected in each layer to find the final parameters. (We can consider $W_0(l) = b(l)$).

$$w^* = \arg \min_w L(w, x, y) \quad (4.31)$$

$$w_{t+1} = w_t - \eta \frac{\partial L(w, x, y)}{\partial w} \quad (4.32)$$

When applying the backpropagation method, we need to calculate two-round; one is feed-forward and other back forward through the network to find the derivative of the Loss function.

First, we need to design the neural network, feed-forward neural network fully connected. We choose some hyperparameter of the model as follow:

- 1) We can select the Loss function to depend on the problem, we can choose some Loss functions like below:

$$L(w) = \frac{1}{2N} \sum_{i=0}^N (y_i - \hat{y})^2 = \frac{1}{2N} \sum_{i=0}^N (y_i - w_i x_i)^2 \quad (4.33)$$

$$L(w) = - \sum_{i=0}^N (y_i \ln z_i + (1 - y_i) \ln(1 - z_i)) \quad (4.34)$$

- 2) We can choose the activation function in the hidden layer and the output layer. The activation function is can sigmoid, tanh, or ReLU functions, with their derivatives are easy to calculate, for example, we choose sigmoid activation function in this case.

$$a = \sigma(z) = \frac{1}{1 + e^{-z}} \quad (4.35)$$

Sigmoid activation functions $a = \sigma(z)$ this activation function has derivative:

$$a' = \sigma(z) * (1 - \sigma(z)) \quad (4.36)$$

- 3) The number of the hidden layers

For example, L layer, each layer has $(d^L - 1)$ neuron (unit) node for $(d^L - 1)$ weight parameters w_{ij} and one bias b_i .

Suppose $L(W, b, X, Y)$ is the loss function of the problem, we can choose loss function where W and b are all weight matrices between each layer and the deviation of each layer. X , Y are input-output data sets.

An example of a loss function is the Mean Square Error function (MSE), which means the average of the squared error.

$$L(\mathbf{w}) = MSE = \frac{1}{2N} \sum_{i=1}^N (y_i - \hat{y})^2 = \frac{1}{2N} \sum_{i=1}^N (y_i - f(w_i x_i + b_i))^2 \quad (4.37)$$

The loss function $L(\mathbf{w})$ is any loss function, not just the MSE function above.

The activation function can choose the sigmoid function, for example.

First, where input data \mathbf{x} , \mathbf{y} , and initial parameter \mathbf{W} , \mathbf{b} , we can calculate the output \hat{y} with an input x . This step is feed-forward because the calculation is done from the beginning to the end of the network.

$$\mathbf{a}^{(0)} = \mathbf{x} \quad (4.38)$$

$$z_i^{(l)} = w_i^{(l)T} \mathbf{a}^{(l-1)} + b_i^{(l)} \quad (4.39)$$

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)T} \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}, l = 1, 2, \dots, L \quad (4.40)$$

$$\mathbf{a}^{(l)} = f(\mathbf{z}^{(l)}), l = 1, 2, \dots, L \quad (4.41)$$

L is the output layer

$$\mathbf{a}^{(L)} = \hat{y} \quad (4.42)$$

$\mathbf{a}^{(l)} = f(\mathbf{z}^{(l)}) = \sigma(\mathbf{z}^{(l)})$ this is an activation function.

N is the number of data pairs (x, y) in the training data set.

Stochastic Gradient Descent can be used to calculate derivative for weight matrices and biases based on a pair of training x and y points.

According to the above formulas, the direct calculation of this value is extremely complicated because the loss function does not depend directly on the parameter (\mathbf{W}, \mathbf{b}) . The most effective method, called Backpropagation, was used in this case, which calculates the

reverse gradient from the last layer to the first layer. The final layer is calculated first because it is closer to the outputs and loss functions. The gradient calculation of previous layers is based on a common rule called chain rule calculus, which is the derivative of the Loss function.

Stochastic Gradient Descent has a loss function $L(\mathbf{b}, \mathbf{W}, \mathbf{x}_i, y_i)$ the derivative of the loss function follows only one component (x_i, y_i) , the weight matrix of the final layer can calculate:

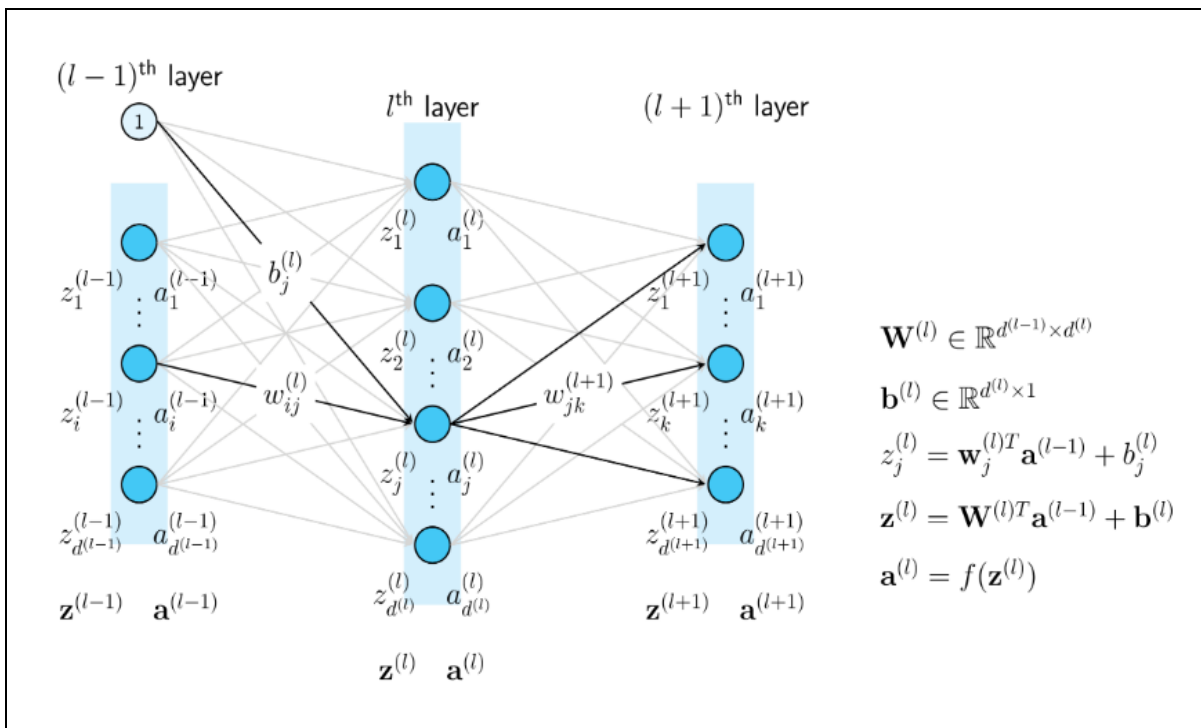


Figure 4.21 Backpropagation demonstration
Taken from Vu Huu Tiep (2018, p. 201)

$L = l+1$: output layer, L in this layer we have $a^{l+1} = \hat{y}$

The last hidden layer is $l = L-1$. Base on the chain rule, the derivatives of each parameter was calculated from the output layer backward to $L-1, L-2, \dots$, and the input layer.

$$L(w) = MSE = \frac{1}{2} (y_i - \hat{y})^2 \tag{4.43}$$

Each layer we need recalculate $\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial a_j^l}$ for using this value for the next step:

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial a_j^{l+1}} = \frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial \hat{y}}$$

We apply chain rule derivatives:

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial w_{jk}^{l+1}} = \frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial a_j^{l+1}} \frac{\partial a_j^{l+1}}{\partial z_j^{l+1}} \frac{\partial z_j^{l+1}}{\partial w_{jk}^{l+1}} \quad (4.44)$$

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial w_{jk}^{l+1}} = \frac{\partial (\frac{1}{2} (y_i - \hat{y})^2)}{\partial \hat{y}} f'(z) a_j^l \quad (4.45)$$

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial w_{jk}^{l+1}} = (\hat{y} - y_i) f'(z) a_j^l \quad (4.46)$$

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial b_{ij}^l} = (\hat{y} - y_i) f'(z) \quad (4.47)$$

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial a_j^l} = (\hat{y} - y_i) f'(z) w_{jk}^{l+1} \quad (4.48)$$

We continuous calculate for the second last hidden layer is $l = L-2$.

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial w_{jk}^l} = \frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial a_j^l} \frac{\partial a_j^l}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{jk}^l} \quad (4.49)$$

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial w_{jk}^l} = \frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial a_j^l} f'(z) a_j^l \quad (4.50)$$

Using the value from equation (4-48)

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial w_{jk}^l} = ((\hat{y} - y_i) f'(z) w_{jk}^{l+1}) f'(z) a_j^l \quad (4.51)$$

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial b_{ij}^l} = ((\hat{y} - y_i) f'(z) w_{jk}^{l+1}) f'(z) \quad (4.52)$$

$$\frac{\partial L(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i)}{\partial a_j^{l-1}} = ((\hat{y} - y_i) f'(z) w_{jk}^{l+1}) f'(z) w_{jk}^l \quad (4.53)$$

In practice, it is common to combine backpropagation with a learning algorithm such as stochastic gradient descent. In this method, we compute the gradient for many training samples. In particular, given m training samples in a mini-batch, the following algorithm implements a gradient descent learning step based on that mini-batch as below:

Step 1 Input a set of training examples

For each training example: Set the corresponding input activation $a^{x,l}$, and perform the following steps:

Step 2 Feedforward

For each $l=2, 3, \dots, L$ compute

$$z^{x,l} = w^l a^{x,l-1} + b^l \text{ and } a^{x,l} = f(z^{x,l}).$$

Step 3 Backpropagate

Output error $\delta^{x,L}$:

Compute the vector

$$\delta^{x,L} = \nabla_a L_x \odot f'(z^{x,L})$$

Backpropagate the error:

For each $l=L-1, L-2, L-3, \dots, 3, 2$ compute vectors

$$\delta^{x,l} = ((w^{l+1})^T \delta^{x,l+1}) \odot f'(z^{x,l})$$

Step 4 Gradient of the Loss function with respect to w, b parameters

$$\frac{\partial L_x}{\partial w^{x,l}} = \delta^{x,l} (a^{x,l-1})^T$$

$$\frac{\partial L_x}{\partial b^{x,l}} = \delta^{x,l}$$

Step 5 Gradient descent

For each $l=L, L-1, \dots$, update the weights according to the rule:

$$w^* = \underset{w}{\operatorname{arg\,min}} L(w, x, y) \quad (4.54)$$

$$w_{t+1} = w_t - \eta(\delta^{x,l}(a^{x,l-1})^T) \quad (4.55)$$

and the biases according to the rule

$$b_{t+1} = b_t - \eta\delta^{x,l} \quad (4.56)$$

With learning rate η (for example $\eta = 0,001$) we can calculate follow gradient descent method.

CHAPTER 5

APPLY DEMAND FORECAST FOR ETS BUILDING USING NEURAL NETWORK

Before building the demand forecast model applies in the district building, We review the checklist Machine Learning projects Appendix B. Machine Learning Project Checklist book [38]. There are eight main steps:

- 1) Overview demand forecast and review existing method and solution.
- 2) Collect the data preparation for the solution.
- 3) Explore and analyze the data to obtain insights.
- 4) Prepare the data to expose better the underlying data patterns to implement Machine Learning algorithms.
- 5) Examine many different models and short-list the best ones.
- 6) Fine-tune the models and assemble them into a great solution.
- 7) Present Machine Learning solution.
- 8) Launch, monitor, upgrade and maintain the machine learning system.

Systematically, we will go through two important parts, first is work with data, and the second is work with the neural network model. The main concern about data:

- 1) Is there any relation between forecast input and output?
- 2) Can be connected (available), and stable data source?
- 3) Accurate data, data sources, data measurement points, missing data, abnormal data, duplicate data, and outlier data?

We have three data sources:

- 1) Historical electrical energy data: the form of a smart metering system for measuring electricity or managing energy distribution system. Notice the energy power value (kWh) connected from the smart meter can convert to capacity power value (kW).
- 2) Calendar data: for example, each university, we have an academic calendar, which has common events and school events. For the official holiday in university, it will be viewed as the weekend.

3) Weather data source from the government website: <https://weather.gc.ca/>

Data can collect from 2013, in these two years, data in 2015 - 2016 will consider as input data for training, validation, and testing model.

5.1 Data pre-processing

Before analysis and apply for the neural network model, data need to pre-process.

Feature scaling: standardize or normalize features give all value of data in the range [-1 to 1] or [0 to 1] base on min-max standard technique, or standardization technique will be applied in MATLAB toolbox or Python code.

5.1.1 Extract feature data

It is analyzing data based on knowledge and experience, based on collected data to quantify input variables affecting predictive model output results. It is possible to use the Pearson correlation coefficient, R square, to evaluate the linear relationship. Note that the variables need to be independent of each other. If the variables are dependent, it is not the model's characteristic. Therefore, it does not bring benefit value for the model after the training process.

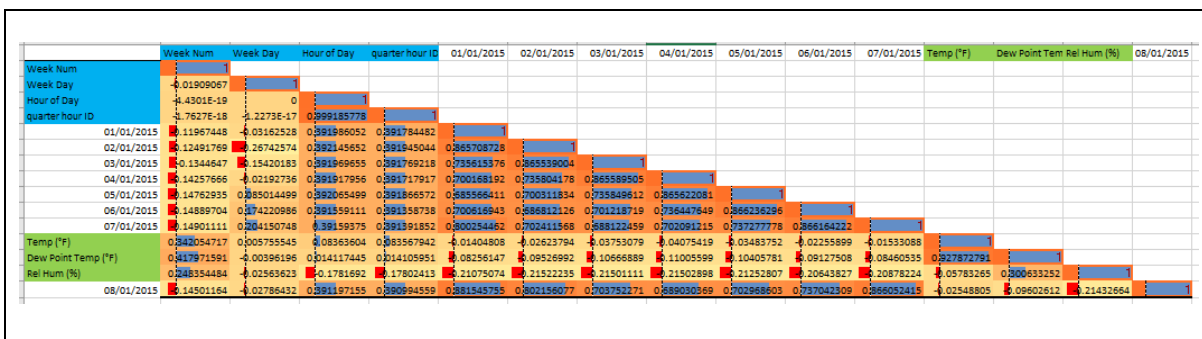


Figure 5.1 Pearson correlation coefficient of 11 variables with one output

For model 24 hours ahead, 11 variables input and 1 output were selected:

The historical power variable is the power data of Day-7, Day-3, Day-2, Day-1.

The weather variables are, temperature, dew point temperature, and humidity.

The calendar variables are Week of the year, Day of Week, Hour of the Day, and Quarter (15 minutes).

For model 24 hours ahead, 7 variable inputs (the power data of Day-7, Day-2, Day-1, Week of the year, Day of Week, Hour of the Day, and temperature) and 1 output were selected.

For model two hours ahead, 14 variable inputs and 8 outputs were selected:

The historical power variable is the power data of Day-7, Day-1 Hour-15, Hour-30, Hour-45, Hour-60, Hour-75, and Hour-90.

The weather variables are Temp, Dew point Temp, and Humidity.

The calendar variables are Day of Week, Hour of the Day, and Quarter (15 minutes).

5.1.2 Detection and Handling of data missing and duplicate data

The duplicated and missing data can find base timestamp data. We can check the timestamp variable. Use it through the time stamp because the time series intervals are equal. We need to remove the duplicate data and fulfill missing data. The missing data we can choose the same technique clean up outlier data technique presented as follow.

5.1.3 Clean up outlier

Handle a small number of abnormal data and handle abnormal phenomena such as electric outage planned, or circuit breaker trip out. Some problems can ignore and remove this data, such as vision processing computer or language translator machine, but for the demand forecast problem, we should choose an alternative with similar data.

- It is possible to prioritize alternative data such as backup data; measurements from the main smart meter will be able to be replaced by sum up branches smart meter.
- Data from previous and subsequent cycles.
- Moving average weight.
- A similar date.

It is also not too important to find the best way to replace this data. The primary purpose of reducing noise, biases from abnormal data that we have known in advance so as not to affect the training process. In this case, we use the median method or the similar day method to clean up outliers.

5.1.4 Adjustment Daylight saving time data

Every year Daylight saving time is adjusted from 2:00 AM to 3:00 AM on a Sunday in the second week of March and decreased from 2:00 AM to 1:00 AM in the first week of November. Detail adjust Daylight saving time are shown in the table below:

Table 5.1 Daylight saving time from 2013-2019
Adapted from Timeanddate (2019)
www: <https://www.timeanddate.com/time/change/canada?year=2019>

	The second week of March 2h00 AM forward to 3h00 AM	The first week of November 2h00 AM backward to 1h00 AM
2013	10-March	03-November
2014	09-March	02-November
2015	08-March	01-November
2016	13-March	06-November
2017	12-March	05-November
2018	11-March	04-November
2019	10-March	03-November

Because two periods (2h00-3h00 AM and 2h00-1h00 AM) is an off-peak period, the weather and power data can keep the same and average value respectively in two periods for simple. After this period, data will be synchronized with the daylight-saving timestamp.

The historical power data of the district building in downtown Montreal are collected from the metering system and utilized in this thesis. The district building including seven buildings with several functions (student residential, commercial center, office, and research and education buildings). Therefore, the power load profile they have represented diversity consumption. However, some buildings are small consumption then we can group into five main buildings.

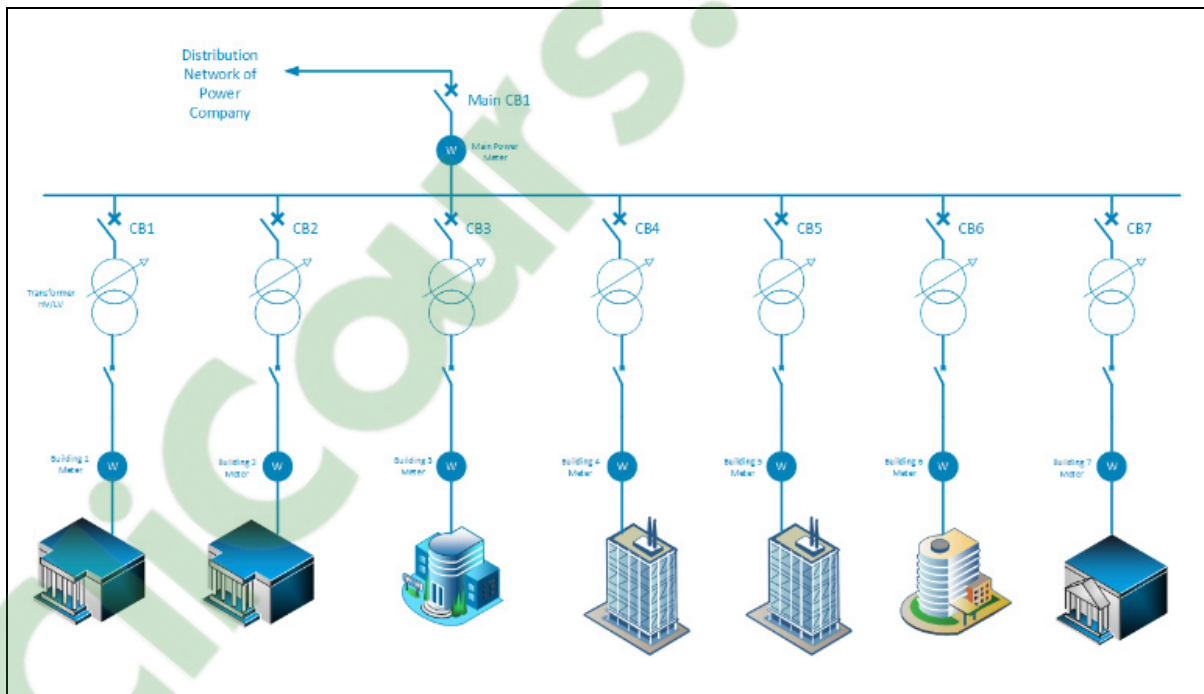


Figure 5.2 Example of a district buildings

It can be seen in Figure 5.3 below that the loads change significantly by season throughout the year and are highly random between days in a typical year. In this case, the total used capacity of buildings is subscribed with a capacity of 5MW. It means the electric power bill has two components (capacity and energy). The capacity value billed is determined by retaining the highest value in kilowatts between the maximum actual energy demand and the registered capacity (5MW). In Figure 5.3 below, the bold and dashed line describes the

registered power level. Figure 5.3 also shows that in winter and summer of the year, the power demand of buildings is often higher than the capacity subscribed (5MW).

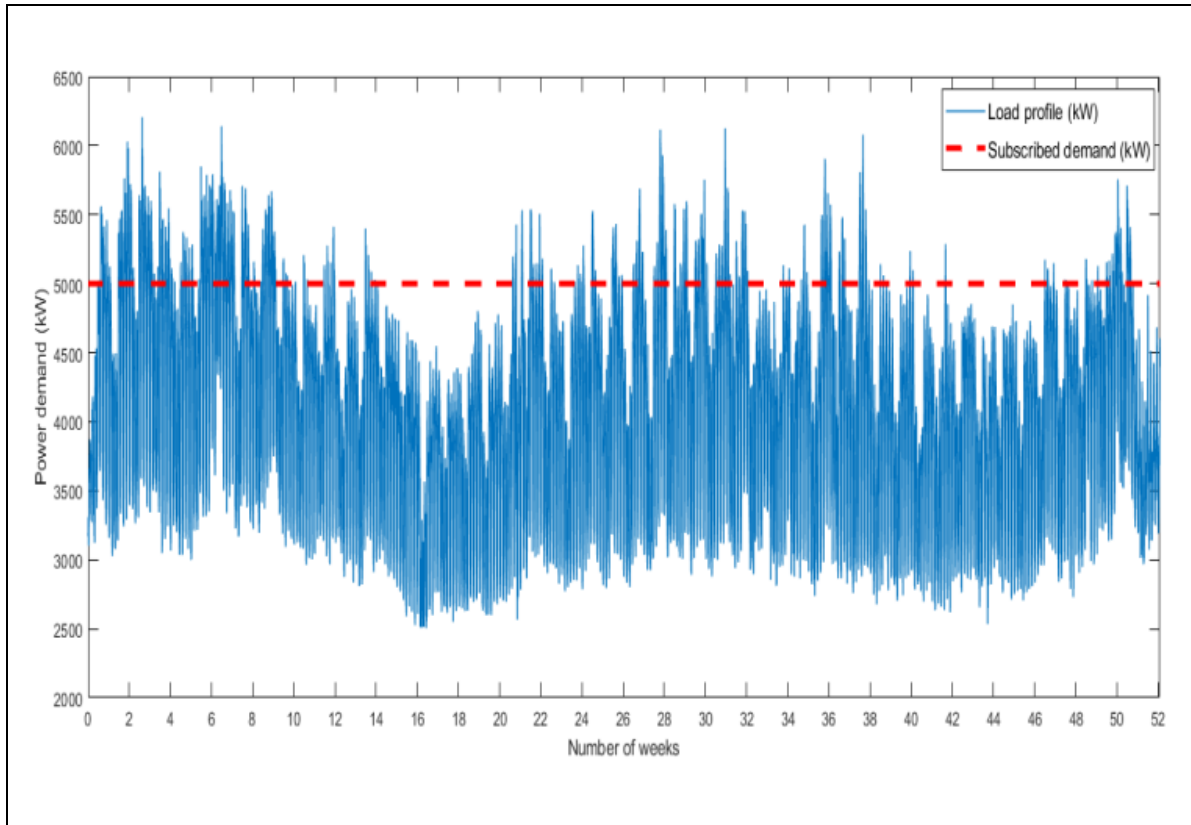


Figure 5.3 Electricity demand in the district building in one typical year

The electricity consumers were applying Hydro-Québec's LG tariff. The electricity consumers require to try to make an accurate demand forecast special in the peak period to optimize electricity use. Because if used more than 110% of the subscribed capacity, Hydro-Québec will charge an additional daily penalty of \$ 7.11 per each kW [54]. Therefore, the task of forecasting electricity demand in the peak period is very important for electricity consumers.

Figure 5.4 displays power over the subscribed power for four years. It is shown clearly that the peak demand has a trend towards increasing every year.

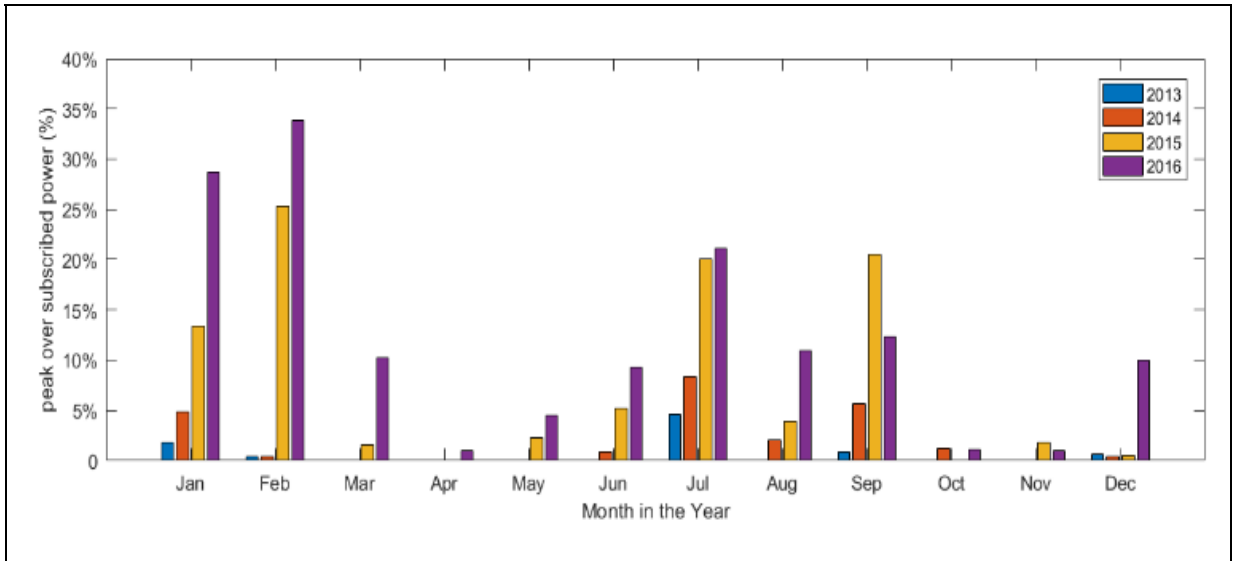


Figure 5.4 Percentage of peak power demand over than subscribed from 2013 to 2016

Figure 5.5 below displays the typical weekly load variations in each season. The electricity demand using for cooling and heating accounts for a significant proportion, which is reflected in the higher electrical capacity in summer and winter during weekdays. Besides, the trend changes are similar between working days and weekends for all seasons in years.

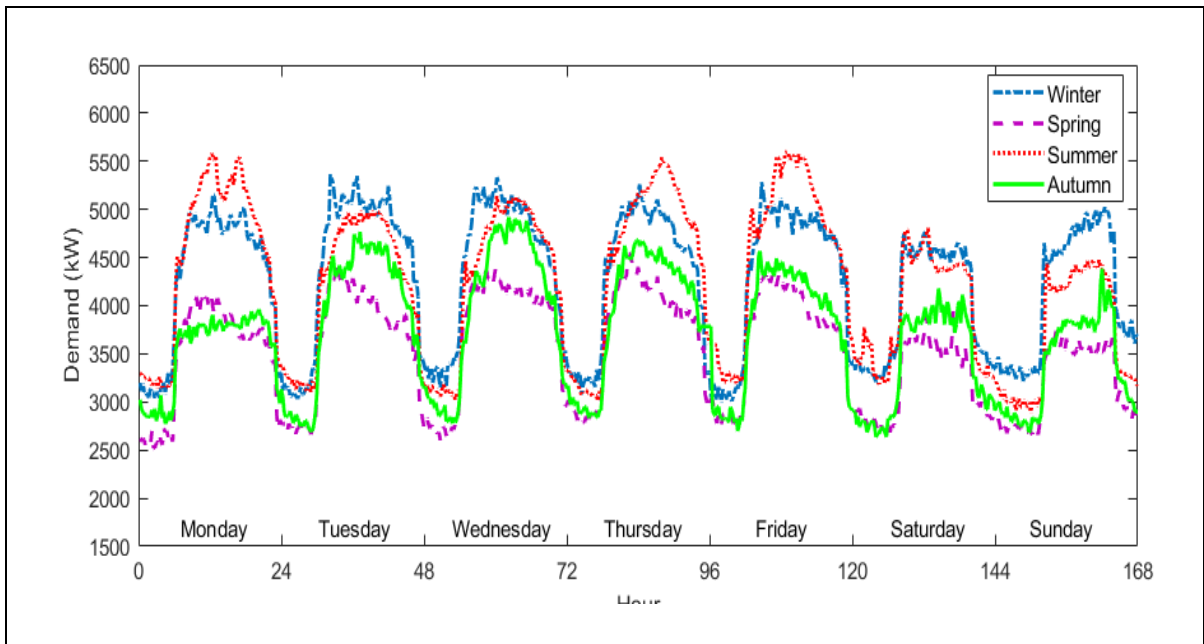
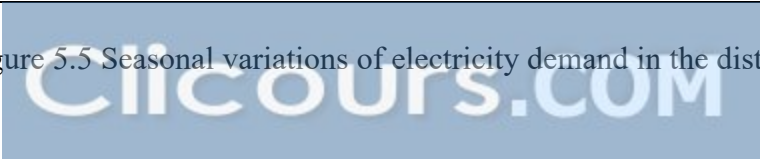


Figure 5.5 Seasonal variations of electricity demand in the district building in a week



There is a modern power management system for the whole buildings. The main electricity meters used to pay bills with the power company. Individual buildings also have sub-meters. The meters are all set to provide power measurements in 15 minutes period and much more other information.

Figure 5.6 shows the separate load of the five buildings. Buildings 1 and 2 are entirely institutional functions, while buildings 3 and 4 contain residential and commercial loads.

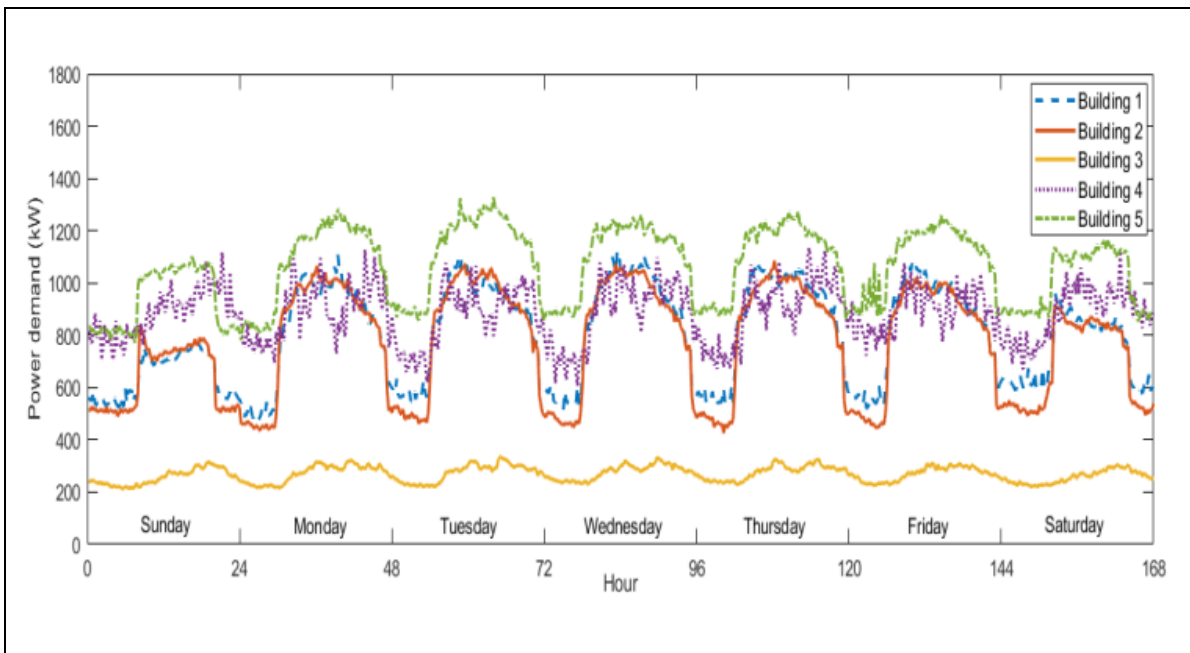


Figure 5.6 Weekly load profile for the five buildings

The load profile of each building reflects the diversity of electricity use. Each building has a different load profile shape. For example, buildings 1 and 2 have a significant change in their electricity use for a period of 6 AM - 8 AM. Buildings 1 and 2 also show many differences between weekends and working days. In contrast, the change between weekends and working days does not occur in building 3. Building 4 shows fluctuations in the day. Many buildings display a considerable change, making forecasting more difficult.

5.2 Methodological approach

Power load forecasting is a complicated and challenging work. Based on chapters 1, 2, 3, and 4, the neural network was purposed neural network model to adapt to this task. MATLAB is an academic power software. The neural network builds in MATLAB, easy to connect with another MATLAB module via Simulink/MATLAB. Another program is Python, which is an open source code, has a large community user, and supports convenient libraries. This section will present an overview of four neural network models to apply to the next 24h00 load forecast. Two models of Feed Forward Neural Network using MATLAB toolbox and two models using Deep Neural Network and Long Short-Term Memory in Python language.

The model is implemented for the next 24-hour forecast and the next 2 hours forecast with a 15-minute resolution. The model is designed to predict each building and the total electricity of district buildings. The final section will also cover the effectiveness of combining forecasting models.

First, performance measures were selected to evaluate each model.

5.2.1 Performance measures

There are some standard metrics to evaluate the performance of neural network algorithms. The mean absolute error efficiency (MAPE) and the mean square root error (RMSE) were calculated for different neural network architectures. These values give a general idea of the difference in predicted values and targets.

$$RMSE = \frac{1}{n} \sqrt{\sum_{i=1}^n (y_{p,i} - y_{data,i})^2}$$

$$MAPE(\%) = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_{p,i} - y_{data,i}}{y_{data,i}} \right| \times 100$$

Where $y_{p,i}$ and $y_{data,i}$ represent respectively, the predicted power demand at the time i and the actual power consumption at the time i .

RMSE is sensitive to significant errors, while MAPE is less sensitive and easier to observe because of the same measurement value.

5.2.2 Neural network model

The model is implemented for the next 24-hour forecast task and the next 2 hours with a 15-minute resolution. Data save in files with *.csv format, Matlab, and Python can easily read the data for training or testing.

The model is designed to predict each building and the total electricity of buildings in the district building.

The Neural network-based architecture proposed a neural network model was summarize in the table as follow:

Table 5.2 Four neural network models and their hyperparameters

Characters	Model 1	Model 2	Model 3	Model 4
Langue program	MATLAB toolbox	MATLAB toolbox	Python	Python
Neural network structure	Feed-forward One hidden layer Fully connected	Feed-forward One hidden layer Fully connected	Feed-forward Deepthree hidden layers Fully connected	Feed-forward LSTM Three hidden layers Drop-out – 20%
Neuron in hidden layer	24	24	15/15/15	30/20/20

Characters	Model 1	Model 2	Model 3	Model 4
The activation function in the hidden layer	Sigmoid function	Sigmoid function	ReLU-Rectified Linear Unit	ReLU
Loss function performance	Mean square error (MSE)	Mean square error (MSE)	Mean square error (MSE) or Mean Absolute Error (MAE)	Mean square error (MSE) or Mean Absolute Error (MAE)
Algorithm	Levenberg-Marquardt	Bayesian Regularization	Adam - Adaptive Moment Estimation	Adam - Adaptive Moment Estimation
Calculates method.	Backpropagation Gradient Descend	Backpropagation Gradient Descend	Backpropagation Gradient Descend	Backpropagation Gradient Descend
Training data set rate	Training 70%	Training 80%	Training 90%	Training 90%
Avoid over-fitting	Stop early	Regularization	Stop early	Stop early Drop-out
Epoch	300/1000	500/1000	180	180
Time training	Fast Less than 5 minutes	Average Less than 10 minutes	Fast Less than 5 minutes	Slow Less than 25 minutes

5.2.3 Backpropagation neural network based on Levenberg Marquardt algorithm

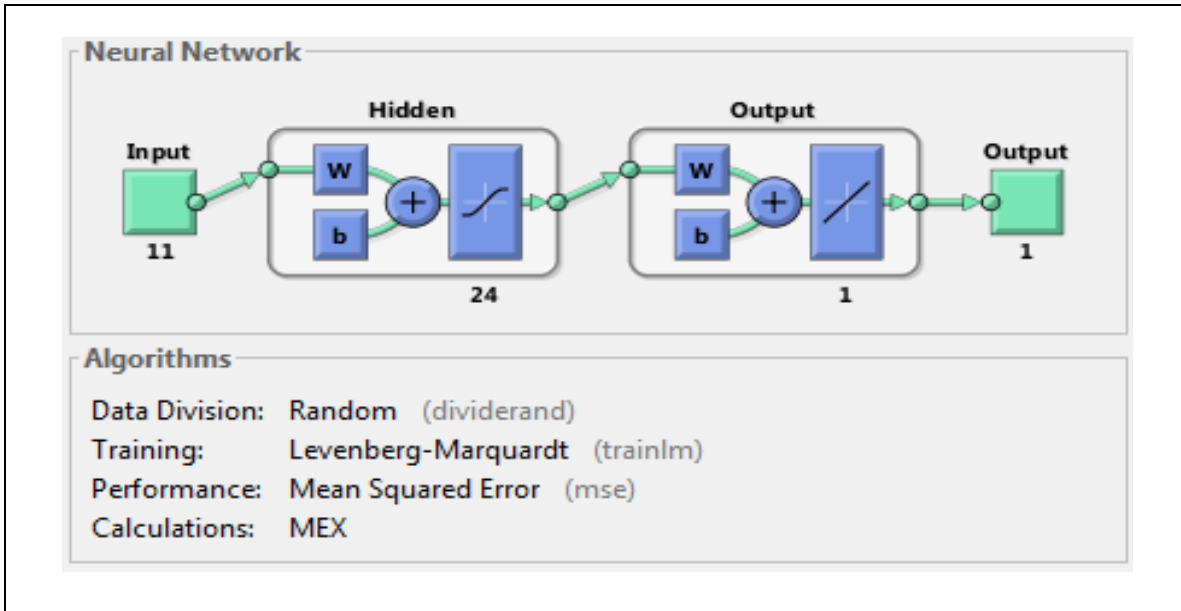


Figure 5.7 Model neural network Levenberg-Marquardt algorithm in MATLAB

5.2.4 Backpropagation neural network based on Bayesian regularization

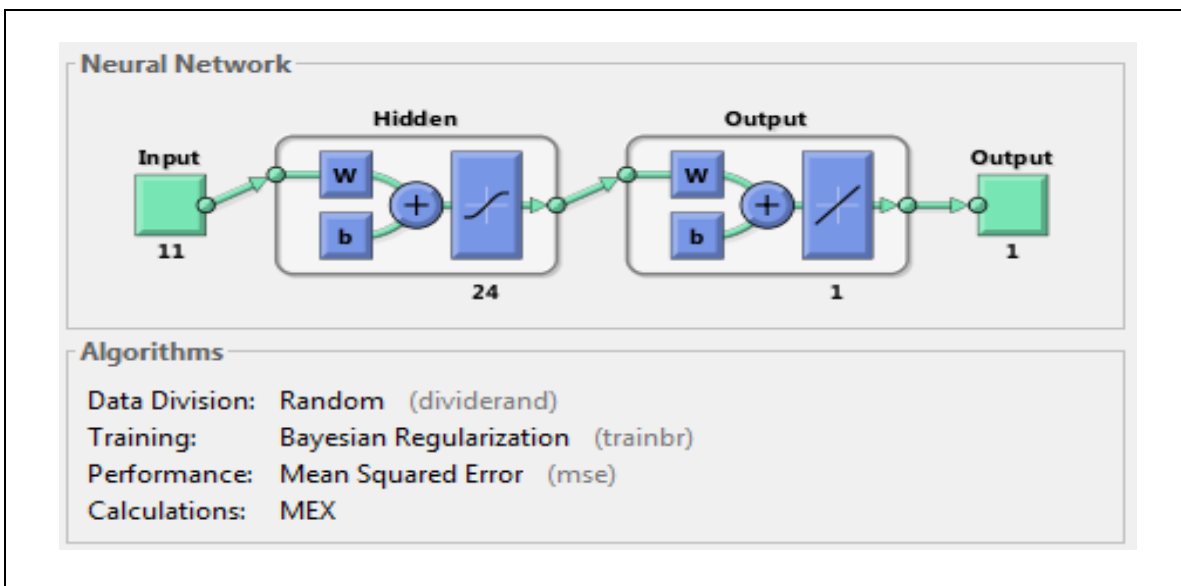


Figure 5.8 Model neural network Bayesian regularization algorithm in MATLAB

5.2.5 Deep learning neural network

The neural network, initially only adding one hidden layer, can solve many complicated non-linear problems with relatively high accuracy. Adding more hidden layers or increasing the number of neurons in the hidden layer will often help the ability of the network training significant more powerful, being able to learn complicated issues. However, it is often possible to have over-fitting problems.

Deep learning is originated from the neural network, just a small industry of machine learning. However, it has grown up rapidly in recent years. With some challenging problems, when applying deep learning, there is a big jump. The main reason for achieving such results is that the computing power of computers has increased rapidly, and there is a massive amount of data from the internet. Deep learning often has many hidden layers to be able to learn complex data (if the data is vast, it can be limited to overfitting above).

The new Backpropagation method, the ReLU activation function, and some new Gradient Descent methods such as ADAM also help to train the network quickly and create accuracy results for Deep learning applications.

Two influential development groups are CNN for image processing and RNN - LSTM to solve natural language problems, automatic translation of languages, etc.

LSTM is an ANN structure that allows us to remember previous consecutive information rather than just only input parameters to understand the characteristics of information better. Using LSTM can bring effects like how we see a movie. We can understand the context better if we see some previous scenes, not just one present scene. LSTM architecture is very successful in translating different languages because it can interpret a phrase of words instead of just an individual word. It is also often applied in time series problems.

LSTM is also often studied with time series problems because of the ability to customize modeling time series data. We can see a detailed LSTM model in figure 5-9 below:

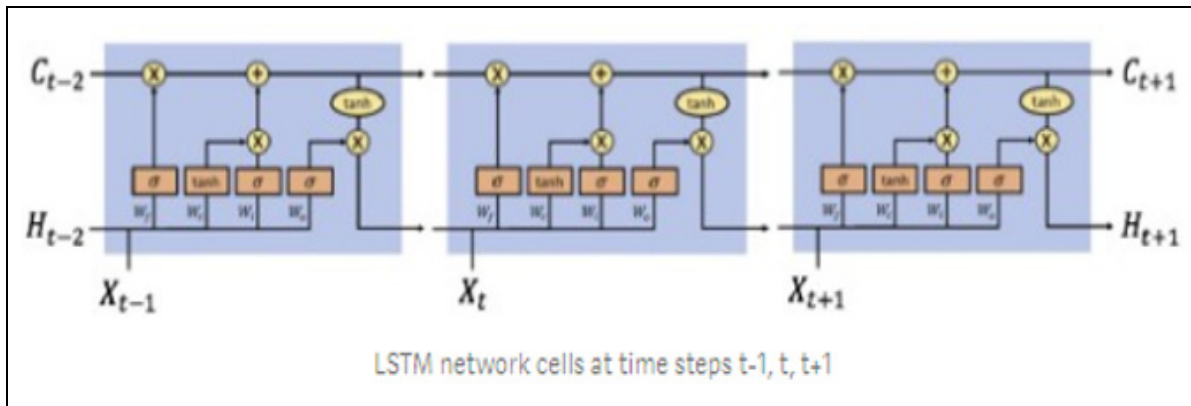


Figure 5.9 LSTM neural network

Taken from Arbel (2018)

www: <https://medium.com/datadriveninvestor/how-do-lstm-networks-solve-the-problem-of-vanishing-gradients-a6784971a577>

(Retrieved from website on February 27, 2020)

The power load prediction problem presented in this thesis is also built with two more deep learning models. The first model keeps ANN feed-forward structure with three hidden layers. The second is increasing the number of hidden layers with three more layers and using the LSTM structure. The model is implemented in Python with related libraries.

We can choose the mean square error (MSE) or mean absolute error (MAE) when the training process to minimize the loss function. We can choose the mean square error (MSE) or mean absolute error (MAE) when the training process to minimize the loss function. In this case, the controlling parameters are the Mean Square Error. It can work both in Matlab and Python. In this case, the controlling parameters are the Mean Square Error.

The results were shown in Table 5.6; the result explained that the result is almost equivalent to the simple ANN network from MATLAB Toolbox. In addition, having more forecasting models helps more tools to evaluate and analyze load forecasting results. The result of the one-week forecasts test set is shown below:

Table 5.3 The result for 24 hours ahead with four neural network models in a typical week

	Mean forecast value (kWh)	Mean Absolute Error (kWh)	Mean Absolute Percentage Error
Actual demand	4242.14		
ML	4240.52	101.36	2.48%
BR	4220.00	107.45	2.58%
Deep NN	4154.17	143.25	3.42%
LSTM	4183.93	139.73	3.30%
Average Combine 2 models in MATLAB	4230.26	101.78	2.51%
Average Combine 4 models	4197.09	112.27	2.68%

5.3 Forecasting base on Individual building

Predicting the load of the building can also start from the power measurement data of each building. The difference is only the metering data of each building compared to measuring the main meter.

The forecast structure model does not change when using the same structure inputs data with the main meter. The result of the model may be different because the load profile of the buildings is different. If load profiles are stable, then the forecast more accuracy.

Forecasts for individual buildings also can consider as a hierarchy demand forecast. This forecast is more comfortable if we have full data from the smart metering system, we can use this data for back up and crosscheck.

After obtaining the forecast results of each building, it can be aggregated into the overall forecast results of the buildings. The combined results from these predictions, often similar to combine forecast models, are stable and usually smaller than just utilizing metering data of the main meter. The results also summarize the presentation in this thesis in the next chapter.

The forecast for each building also helps to study or research the load profile of each building. The forecast result has many useful implications when consumers and power utilities are applying for the DSM/EE program, in the micro smart grid system.

5.4 Two hours ahead load demand forecast

The forecast for the next two hours comes from the demand for a very short-term load forecast. It is often necessary to forecast more accurately at sometimes of using electricity as during peak hours. Forecasting also takes time to have the next action. For example, turn off charging and turn on the re-storage system. If the forecast too close, then it will reduce forecasting meaning.

Implementing the next two-hour forecast is also done on the MATLAB toolbox, just the input data has changed compared to the next 24-hour forecast model.

The result of the next two-hour forecast period will be eight cycles of 15 minutes. The accuracy is usually reduced, so it can be done rolling to use the hourly forecast (four cycles of 15 minutes). Detailed results are summarized in Chapter 6.

5.5 Combine forecast model

Combining some best models will often perform better than working them individually. Ensemble methods work best when the predictors are as independent of one another as possible. One way to get diverse classifiers is to train those using very different algorithms.

Ensemble forecast enhances the chance that they will make very different types of errors, improving the ensemble's accuracy.

Several methods of combining forecasts that are presented in some documents [33] [46] [30].

If there are many different methods, when they combine can be better. A simple method is to average predictions, not to use weights.

Having multiple methods to choose from also helps users have more options before making a decision.

CHAPTER 6

RESULT AND DISCUSSION

In this chapter, the outcomes of the recommended method and discussion are introduced shown. First, summarily introduce the data set was used. Then the details architectures of our neural network are presented. Lastly, the performance comparison between different features of the method was shown.

The first key task in the development ANN model is to select the data set and measure the performance of the generated models. The two years of data set were used (from the first day in 2015 to the last day in 2016), contained 15-minutes intervals separately at a district building in the Montreal region. For weather information, hourly temperature, dewpoint temperature, and humidity are used to enhance the accuracy of demand district building forecasting. The available dataset was lean data processing before, then is divided randomly into three sets for training, validation, and testing data with 70%, 15%, 15% rate, respectively. We mainly focused on 24 hours-ahead and an hour-ahead load prediction with 15 min forecast step (we extract form two hour ahead forecast). We note that day-ahead and hour-ahead load forecasts are required for peak load shaving in the district buildings. Besides, it helps to manage loads and control distributed resources in the buildings.

The contribution result also was presented in the journal Energy and Building [55].

6.1 Effects of input data and training technique

Table 6.1 displays the outcomes of two training algorithms using for power demand forecasting in a day-ahead with seven inputs. The seven used inputs are related to three power demand data, three calendar data, and one importance of the weather data. Power demand data of the last week, power demand data of the last two days, power demand data of the last day, number of the week in the year, number of the day in the week, the hour of the day, and dry-bulk temperature were selected. For all seasons in two years, it can be seen that the BR algorithm shows better results than the LM algorithm. For two years of data, a MAPE

of 3.9% is achieved for the Bayesian regularization algorithm corresponded to 3.24% of the Levenberg-Marquardt algorithm.

Table 6.1 Performance of learning algorithm for 7 inputs

	Levenberg Marquardt		Bayesian Regularization	
	RMSE (kWh)	MAPE (%)	RMSE (kWh)	MAPE (%)
7 inputs				
Spring	128.23	3.17%	126.12	3.12%
Summer	130.00	3.23%	127.93	3.18%
Autumn	130.76	3.25%	127.44	3.18%
Winter	119.72	2.99%	122.56	3.06%
Average in two years	130.60	3.24%	128.50	3.19%

Table 6.2 illustrates the outcomes of a learning algorithm for forecasting the power demand of the district building for 24 hour-ahead using 11 inputs. In this case, four additional inputs have been added to the neural network model. The four inputs are related to dew point temperature, humidity, quarter-hour number, and power demand data for the last three days. It can be seen that the Bayesian regularization algorithm still performs better than the LM-based algorithm. Moreover, if we compare the MAPE and RMSE for 7 and 11 inputs, we can remark clearly that 11 inputs perform better for all seasons, with a difference of 0.1% for the LM algorithm and 0.14% in the BR algorithm. Figure 6.1 below shows the correlation between the number of epochs of the two training algorithms and the value mean square error. The convergence characteristics indicate that the Bayesian regularization training algorithm converges at the optimal point in 250 epochs.

In contrast, using the Levenberg-Marquardt algorithm, the training takes about 300 epochs to converge optimally. Figure 6.2 indicates the regression of the training and validation data associated with the LM training algorithm, which has a value of 0.97282 and clearly

demonstrates the correlation between network outputs and target values. Figure 6-3 displays the regression of the training and testing data associated with the Bayesian generalization training algorithm. Although few outliers can be seen in Figures 6.2 and 6.3, most training, validation, and testing displays generated results along the line.

Table 6.2 Performance of learning algorithm for 11 inputs

11 inputs	Levenberg Macquardt		Bayesian Regularization	
	RMSE (kWh)	MAPE (%)	RMSE (kWh)	MAPE (%)
Spring	125.22	3.11%	121.51	3.01%
Summer	125.29	3.13%	120.87	3.02%
Autumn	126.59	3.16%	121.80	3.05%
Winter	118.40	2.97%	112.22	2.81%
Average in two years	126,14	3.14%	122,10	3,05%

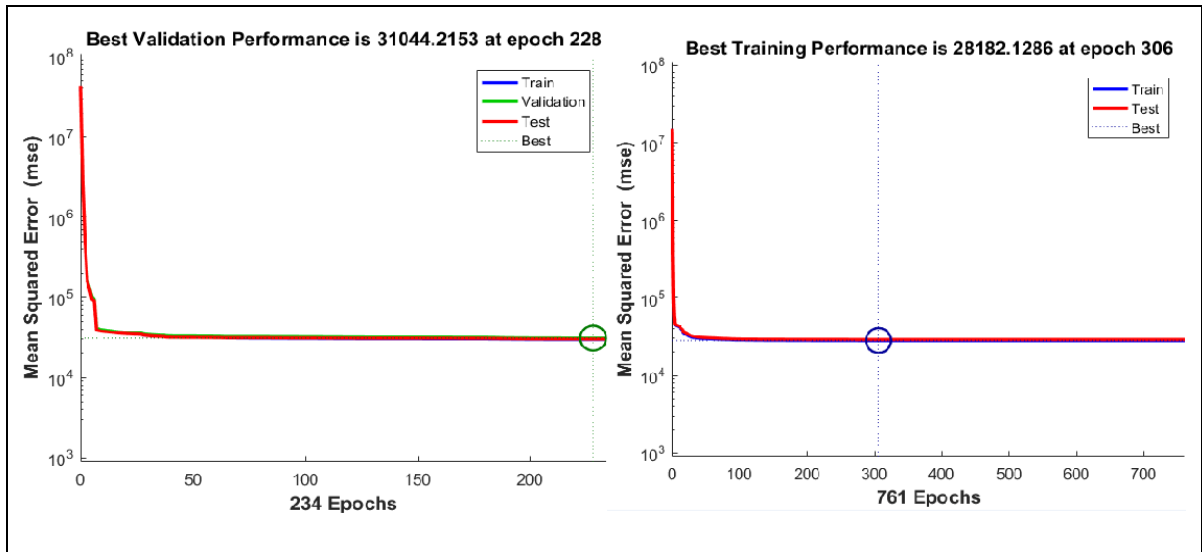


Figure 6.1 The number of epochs vs. mean absolute errors for the LM and BG trained forecasting model

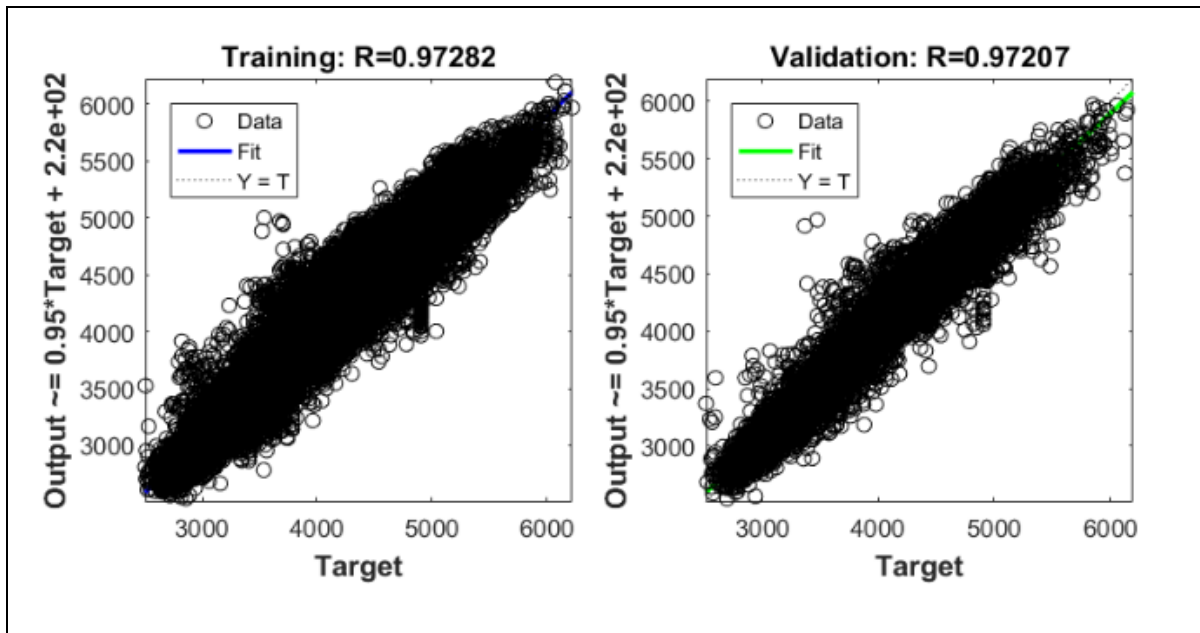


Figure 6.2 Obtained regression for LM backpropagation algorithm

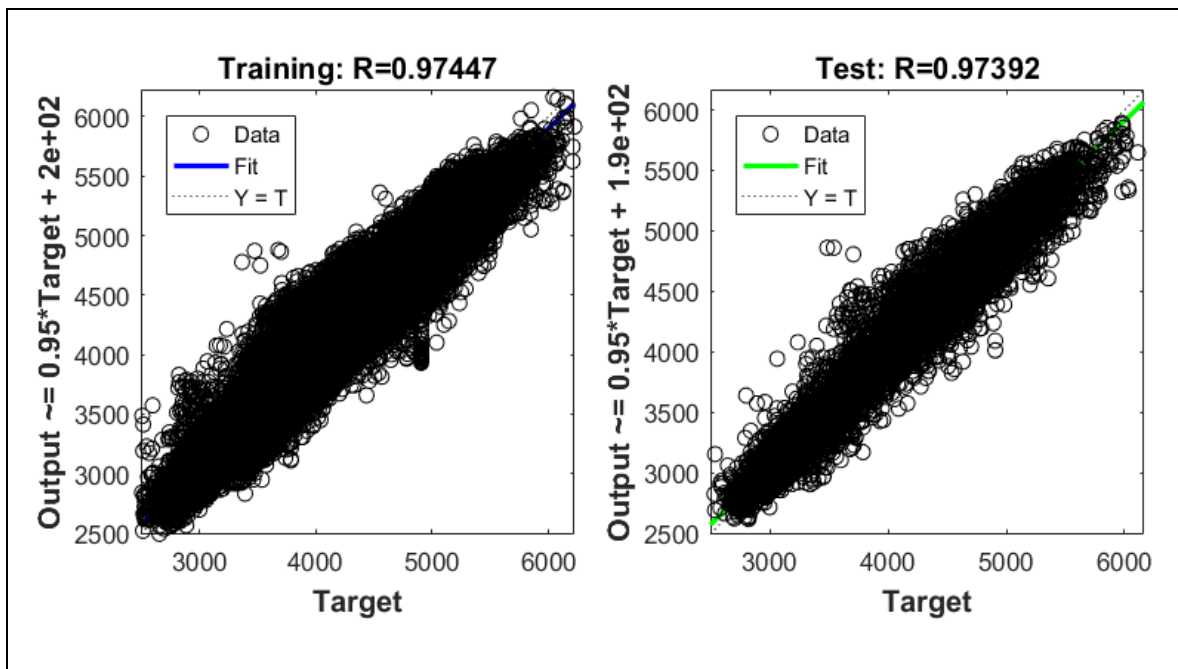


Figure 6.3 Obtained regression for Bayesian regularization

6.2 Effects of number of neurons

To develop the initial neural network structure, include inputs, outputs, and the training algorithm. The number of hidden layers and the number of neurons in the hidden layer is hyper-parameters essential. Regularly, the more hidden layer in the neural network and the more neuron in each hidden layer can be increasing the accuracy of the forecasting model. However, increasing hidden and neuron will come to overfitting problems. In contrast, if just several neurons are chosen in the hidden layer, the training process cannot simulate the complex model.

Therefore, the impact of the number of neurons units on the performance of the neural network model needs to be reviewed. The number of neurons in the hidden layer is varied from 10 to 48. Table 6.3 is presented considerably improves the performance of the neural network model with varied numbers of neurons. In the case of 24 neurons, the RMSE and MAPE is approximately 122 (kW) and 3.05%, respectively. However, the improvement is not significant after 24 neurons. It sometimes is even worst (example for spring season compared with other seasons).

Table 6.3 Performance of learning algorithm with different neurons in the hidden layer

	Number of neurons	RMSE (kW)	MAPE (%)	Mean actual value (kW)
Spring	10	179.50	5.48%	3131.77
	24	111.14	3.45%	
	48	147.55	4.76%	
Summer	10	138.19	3.43%	4022.12
	24	120.87	3.02%	
	48	118.95	2.98%	
Autumn	10	138.96	3.46%	4028.15
	24	121.80	3.05%	
	48	120.19	3.01%	
Winter	10	131.90	3.30%	4068.16
	24	112.22	2.81%	
	48	112.94	2.83%	
Average in two years	10	139.48	3.46%	4037.61
	24	122.11	3.05%	
	48	120.34	3.01%	

6.3 Single versus aggregated buildings prediction

The historical loads for each building were separated and used to train the forecast models. Figure 6-4 shows the actual load and the forecast for 15-min intervals for a week for two typical separate buildings. Good predictions are obtained, in particular from Tuesday to Friday; the neural network model predicts very well the actual load. However, for the

weekend and Monday, the errors are slightly higher. The advantage of load aggregation also is seen in error metrics (Table 6.4). This Table demonstrates the RMSE and MAPE values for three models of prediction. The prediction of power demand in five buildings separate, the prediction of power demand using main meter data and the aggregated prediction of the five buildings are shown. The MAPE for both aggregated prediction and the main meter load prediction is accuracy more than the MAPE recognized for singular buildings. Nevertheless, the aggregating the forecasted loads of all buildings are the best performance.

Table 6.4 Performance of prediction individual building and total

Historical data	RMSE (kW)	MAPE (%)	Mean actual value (target) (kW)
Main Meter			
Total load forecast	126.14	3.14	4037.61
Individual Building			
Building 1	52.80	6.28	873.50
Building 2	38.46	5.00	815.87
Building 3	11.12	3.79	290.03
Building 4	47.83	4.99	969.28
Building 5	41.94	4.08	1088.82
Aggregated forecast	115.7	2.98	4037.99

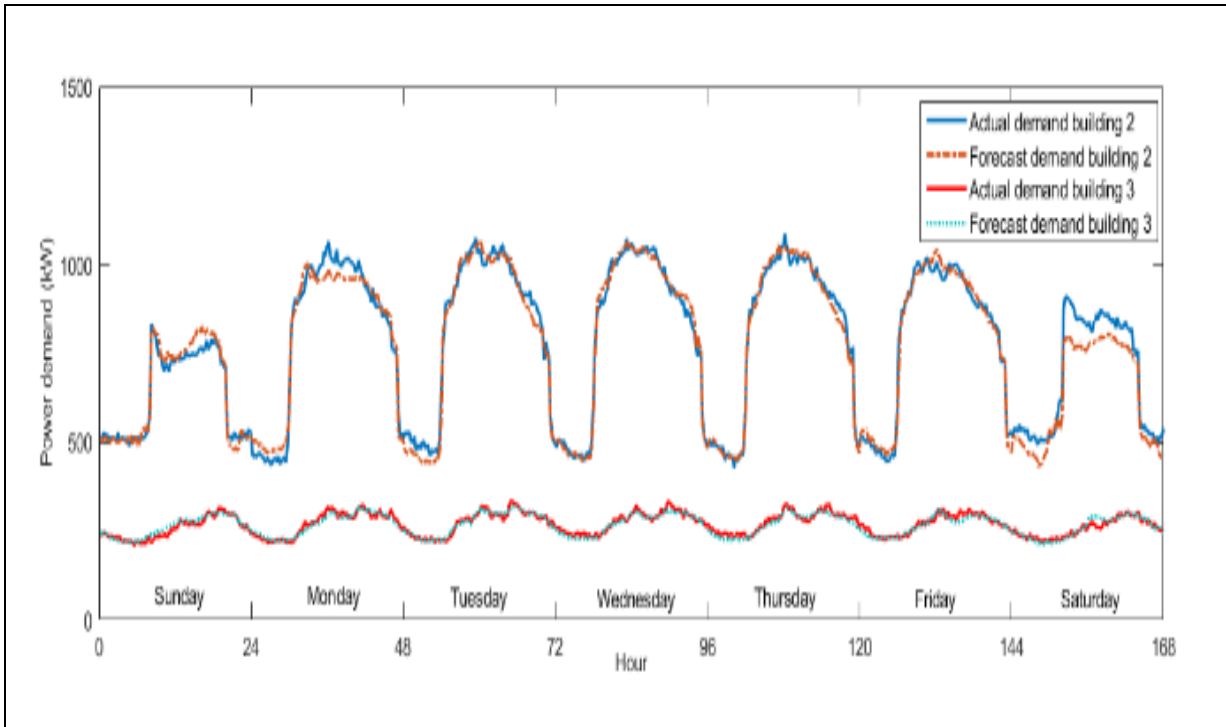


Figure 6.4 Day-ahead forecasts of the individual building in a typical week

6.4 Very short-term load forecast and short-term load forecast

In order to verify the performance of the proposed approach, two load forecast ranges are created: hour-ahead (VSTLF) and day-ahead (STLF) load forecast. Figure 6.5, Figure 6.6, Figure 6.7, and Figure 6.8 show the prediction results with a 15-minute resolution on a typical day in four seasons. In this experiment, 11 inputs and Bayesian regularization training algorithms are selected. As demonstrated in all figures, the performance of the hour-ahead forecast model is better than the day-ahead forecast. This result is mainly due to up to date the power demand in the previous 15 min and the previous 30 minutes, which catch better the change in the load profile, particularly the peak period.

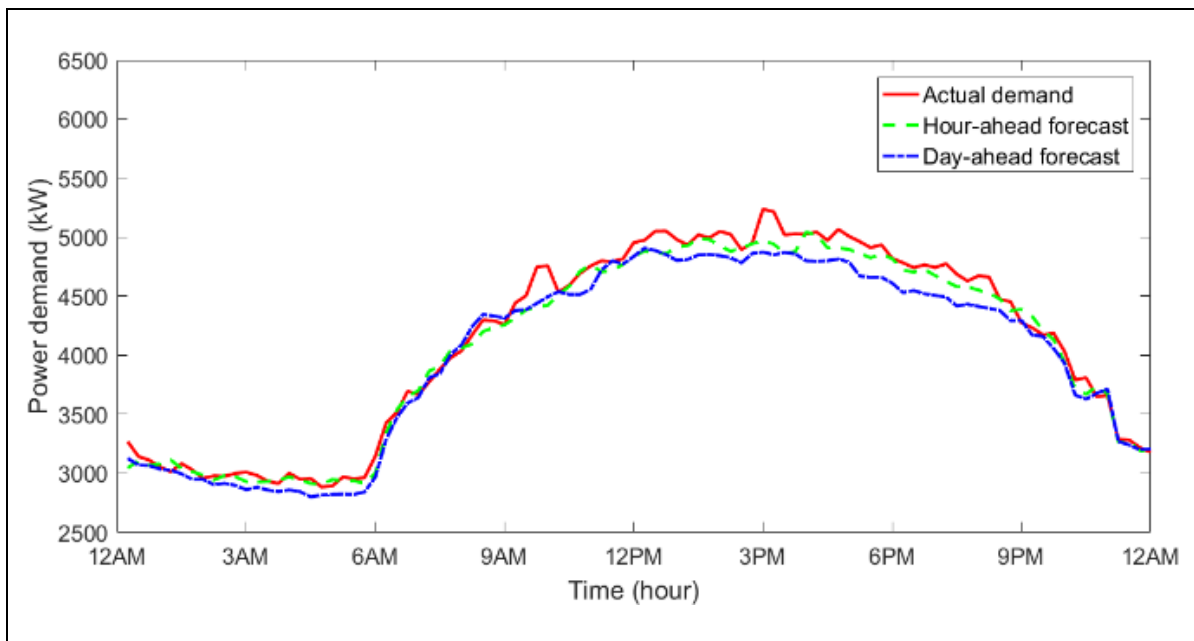


Figure 6.5 Day-ahead versus hour ahead in a typical day in autumn

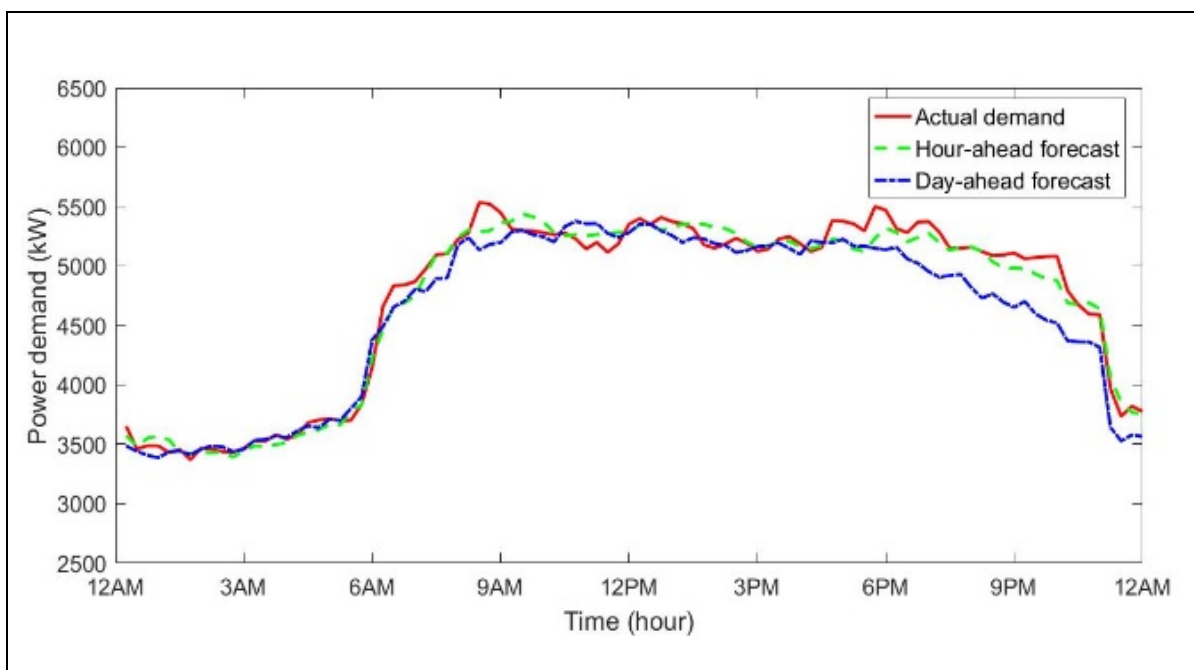


Figure 6.6 Day-ahead versus hour ahead in a typical day in spring

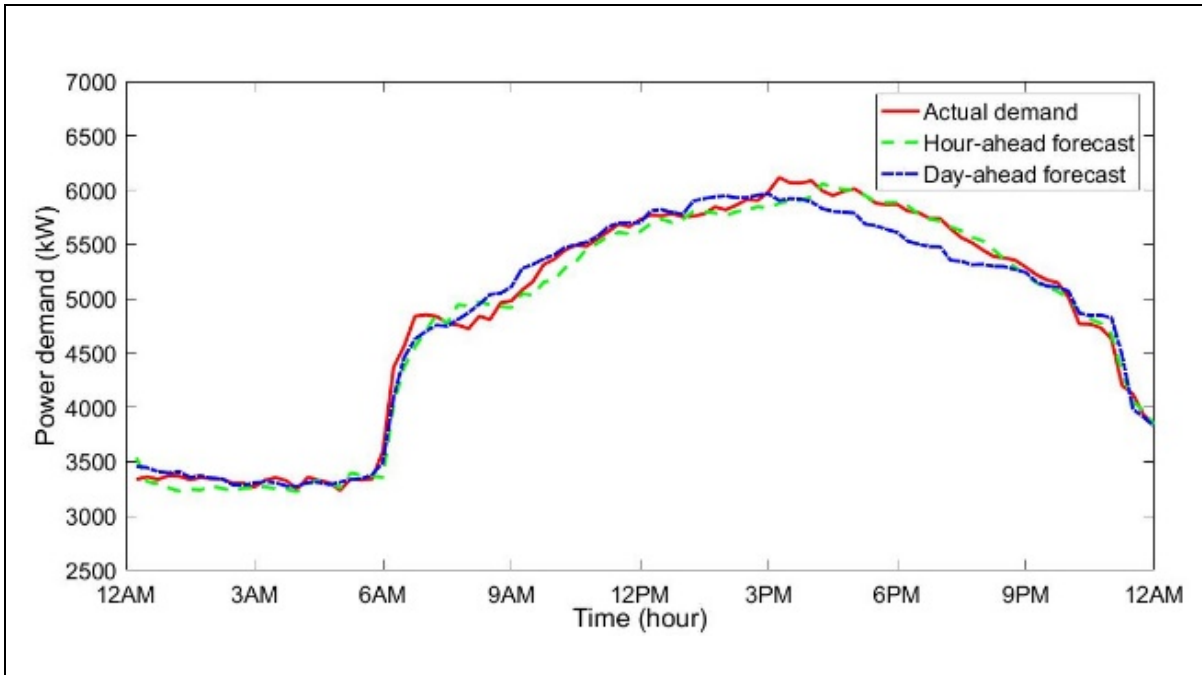


Figure 6.7 Day-ahead versus hour ahead in a typical day in summer

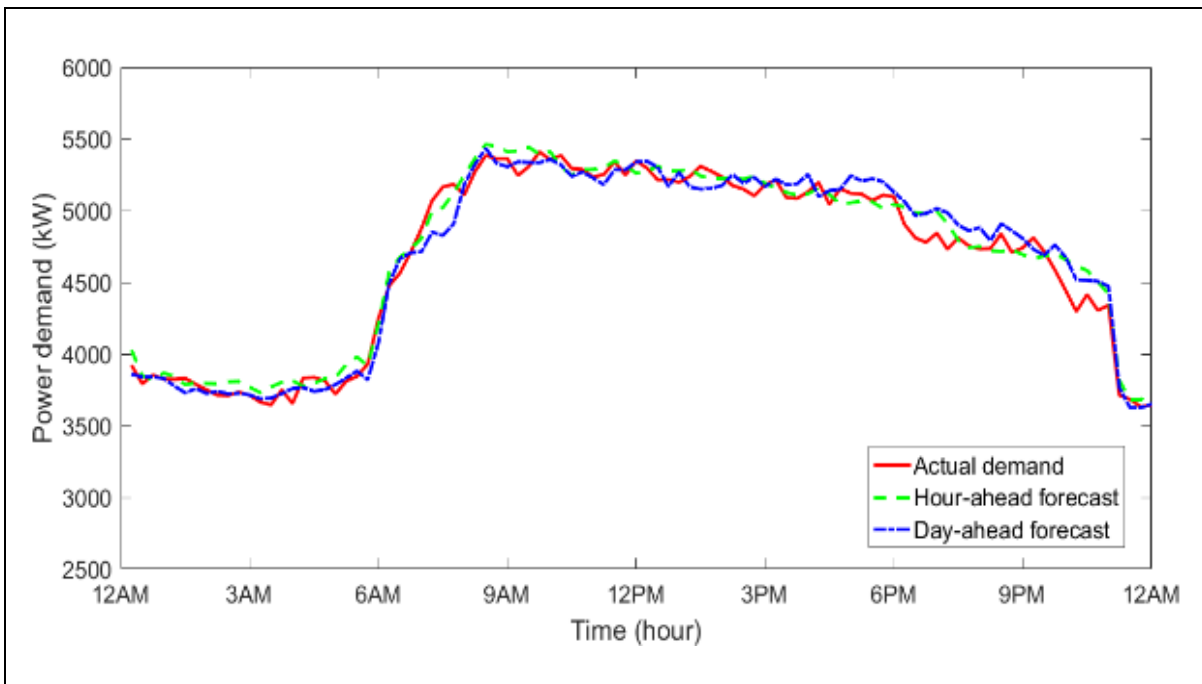


Figure 6.8 Day-ahead versus hour ahead in a typical day in winter

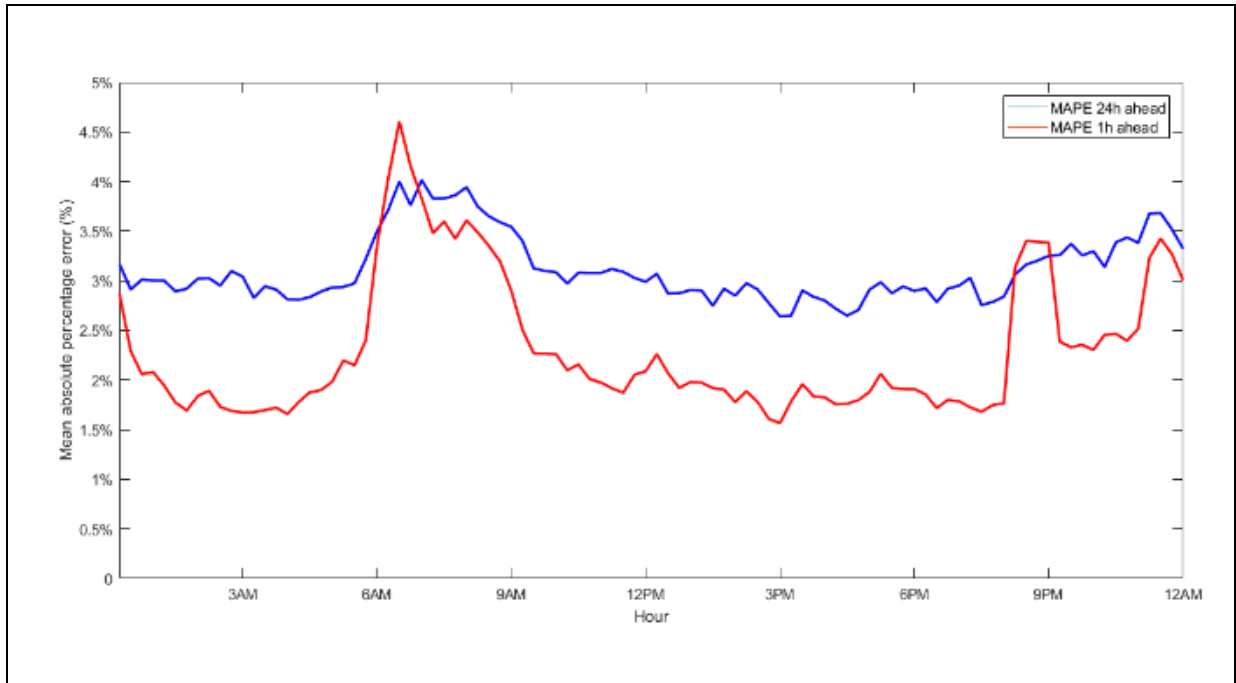


Figure 6.9 The average MAPE for day-ahead and hour-ahead

The average daily MAPE forecast using hour-ahead is lower than forecast using day-ahead, with a difference of about 1%. The differences are shown in Figure 6.9. Both predictions horizons give higher and similar MAPE from 6 AM to 9 AM, reaching a maximum value of 4%. For the rest of the day, hour-ahead is noticeably better for forecasting power demand during a day [55].

6.5 Recommendations for future work

This part of the thesis discusses the results achieved by using a neural network to forecast the electrical load in a district building. This prediction model has used input data like power consumption in the past, time data, and weather-related data. The result is acceptable using the MATLAB Toolbox; this result is easily connected to other simulation software and optimized in MATLAB/Simulink.

The following research contents can be divided into two groups to find a solution to resolve the problem. The first group is linked to data analysis. The skills related to data simulation, data modeling, visualize data, experience, and expert to find suitable inputs using in the

program. The second group is linked with the technique and algorithm for the simulation model.

Two ways also will be related to future work. The first direction is that analysis more details to understand clearly about the forecast model, such as which is the primary system that consumes electricity in the building and how it was operated. How information can be collected to understand more the model and can bring to the model, can we apply for the resident consumer? What is human behaviors impact on electricity demand? For example, the consumer may participate in Demand-side Management (Demand Response/Energy Efficiency) program, and they will use renewable energy resources or electric vehicles?

The second direction is modifying network structure, optimal algorithms, and capabilities computing. For example, to aim to increase the forecasting speed, improving accuracy, we can use some applications like using open-source, iCloud, Big-data, 4G, or 5G technologies. With computer processing speed and the current machine learning algorithm, ANN is quite a capability to solve the load forecasting problem.

The details below are recommended:

6.5.1 The study takes the weight of data over time

Recent data are more important than long past data. However, the data also needs to be large enough to ensure the model can learn from training the neural network and updating recent changes with weight. According to Chapter I in this thesis, from statistics reports, we can show that the electricity consumption of the buildings has reduced significantly compared to the last ten years. Because many energy efficiency devices have been used in recent years. Recent data will become more important than in the past.

6.5.2 Additional data can be added to help improve the model

We can add more data in some case below:

- 1) Update information of data related to thermal systems and electrical systems, because customers may use these two systems to optimize them in peak periods. This input data

- for buildings accounts for a large proportion, so when changing, it will affect the power load forecast results.
- 2) Data related to sunlight intensity (new block building will be applied to the large glass window for saving light form lamp). The intensity of sunlight is also a valuable input to simulate the energy outcome of the photovoltaic rooftop system.
 - 3) Data related to electricity prices, for example, in peak price or off-peak price period, electricity customers applying DSM/EE program will be using electricity different from others.

6.5.3 Improve algorithms and neural network model for load forecast

We can be studied in more detail using Recurrent Neural Network (RNN)/Long Short-Term Memory (LSTM). LSTM has an advantage in solving time-series problems. Assemble forecast models, may study more detail. The combination of each method could be considered, and other options that approach focus only on the peak demand period.

CONCLUSION

In this thesis, the neural network-based model for 24 hours-ahead and an hour-ahead of district building load forecasting was presented detail. The thesis showed how different calendar effects, weather data, neural network structure, and learning algorithms affect the accuracy and the performance of the developed neural network model. For example, the temperature is a valuable input that influences the load forecast accuracy. The proposed neural network model can predict load in district buildings with acceptable accuracy (around 3%). From the results, it can be concluded that the 24 hours ahead forecast, and an hour ahead forecast based on the neural network model will be useful for many purposes. For example, energy management, demand response strategies, peak-shaving, demand-side management program have applied this result. Applies result of demand forecast can support district buildings financially.

APPENDIX I

MATLAB CODE

```
% Solve an Input-Output Fitting problem with a Neural Network
% Script generated by Neural Fitting app
% This script assumes these variables are defined:
%
% Xdata - input data.
% Ydata - target data.

x = Xdata';
t = Ydata';

% Choose a Training Function
% For a list of all training functions type: help ntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. Suitable in low memory situations.
trainFcn = 'trainlm'; % Levenberg-Marquardt backpropagation.

% Create a Fitting Network
hiddenLayerSize = 24;
net = fitnet(hiddenLayerSize,trainFcn);

% Choose Input and Output Pre/Post-Processing Functions
% For a list of all processing functions type: help nprocess
net.input.processFcns = {'removeconstantrows','mapminmax'};
net.output.processFcns = {'removeconstantrows','mapminmax'};

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help ndivision
net.divideFcn = 'dividerand'; % Divide data randomly
net.divideMode = 'sample'; % Divide up every sample
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;

% Choose a Performance Function
% For a list of all performance functions type: help nnperformance
net.performFcn = 'mse'; % Mean Squared Error

% Choose Plot Functions
% For a list of all plot functions type: help nnplot
net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
'plotregression','plotfit'};

% Train the Network
[net,tr] = train(net,x,t);

% Test the Network
y = net(x);
```

```
e = gsubtract(t,y);
performance = perform(net,t,y)

% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
trainPerformance = perform(net,trainTargets,y)
valPerformance = perform(net,valTargets,y)
testPerformance = perform(net,testTargets,y)

% View the Network
view(net)

% Plots
% Uncomment these lines to enable various plots.
%figure, plotperform(tr)
%figure, plottrainstate(tr)
%figure, ploterrhist(e)
%figure, plotregression(t,y)
%figure, plotfit(net,x,t)

% Deployment
% Change the (false) values to (true) to enable the following code blocks.
% See the help for each generation function for more information.
if (false)
% Generate MATLAB function for neural network for application
% deployment in MATLAB scripts or with MATLAB Compiler and Builder
% tools, or simply to examine the calculations your trained neural
% network performs.
genFunction(net, 'myNeuralNetworkFunction');
    y = myNeuralNetworkFunction(x);
end
if (false)
% Generate a matrix-only MATLAB function for neural network code
% generation with MATLAB Coder tools.
genFunction(net, 'myNeuralNetworkFunction', 'MatrixOnly', 'yes');
    y = myNeuralNetworkFunction(x);
end
if (false)
% Generate a Simulink diagram for simulation or deployment with.
% Simulink Coder tools.
gensim(net);
end
```

APPENDIX II

PYTHON CODE

```
import keras
from keras.layers.recurrent import LSTM
from keras.layers import Input, Dense, merge, Dropout
from keras.models import Model
import numpy as np
import sys
import pdb
def create_rnn_model(input_shape, auxiliary_input_dim,
output_dim, loss_func='mae'):
main_input = Input( shape = input_shape)
lstm = LSTM(30, return_sequences = True )(main_input)
lstm = LSTM(20, return_sequences = True )(lstm)
lstm = LSTM(20, return_sequences = False )(lstm)
auxiliary_input = Input( shape = (auxiliary_input_dim,))
auxiliary_dense = Dense(10,
activation='relu')(auxiliary_input)
x = keras.layers.concatenate([lstm, auxiliary_dense])
dense = Dense(15, activation='relu')(x)
#dense = Dropout(0.2)(dense)
#dense = Dense(35, activation='relu')(dense)
dense = Dropout(0.1)(dense)
dense = Dense(15, activation='relu')(dense)
#dense = Dropout(0.2)(dense)
#dense = Dense(48, activation='relu')(dense)
dense = Dropout(0.2)(dense)
prediction = Dense(output_dim, activation='linear')(dense)
model = Model(inputs=[main_input, auxiliary_input],
outputs=prediction)
model.compile(loss=loss_func, optimizer='adam')
return model
def create_dense_model(input_len, output_dim,
loss_func='mae'):
main_input = Input( shape = (input_len,))
dense = Dense(15, activation='relu')(main_input)
#dense = Dropout(0.1)(dense)
dense = Dense(15, activation='relu')(dense)
#dense = Dropout(0.1)(dense)
dense = Dense(15, activation='relu')(dense)
#dense = Dropout(0.1)(dense)
prediction = Dense(output_dim, activation='linear')(dense)
model = Model(inputs=[main_input], outputs=prediction)
```

```

model.compile(loss=loss_func, optimizer='adam')
    return model
def extract_num( word ):
    if word[0] == '':
        word = word[1:]
    if word[-1] == '':
        word = word[:-1]
    return float(word)
def convert_line( line ):
    words = line.split(',')
    numbers = [ extract_num( word ) for word in words ]
    return numbers
def load_simple_csv( file_name ):
    with open( file_name ) as a_file:
        lines = a_file.readlines()
        lines = [ line.strip() for line in lines ]
        lines = [ line for line in lines if len(line) >0 ]
word_list = [ convert_line( line ) for line in lines]
    array = np.asarray(word_list)
#np.random.shuffle( array )
    return array
def train_rnn_model( csv_file, output_file_name ):
    data = load_simple_csv( csv_file )
max_val = np.max(np.abs(data), axis= 0)
np.savetxt('max.txt', max_val)
    #pdb.set_trace()
    data = (data / max_val) - 0.5
    x = data[:, :-1]
    y = data[:, -1]
x_power = x[:, :-1]
x_temp = x[:, -1]
    #x_power = x[:, :-4]
    #x_temp = x[:, -4]
history_len = x_power.shape[1]
single_step_len = 1
x_power = x_power.reshape((-1, history_len, 1))
input_shape = (history_len, 1)
pdb.set_trace()
    model = create_rnn_model( input_shape, 1, 1)
model.fit([x_power, x_temp], y, batch_size=300, epochs=100,
validation_split=0.10)
model.save_weights( output_file_name )

def train_model( csv_file, output_file_name ):
    data = load_simple_csv( csv_file )
max_val = np.max(np.abs(data), axis= 0)

```



```

np.savetxt('max.txt', max_val)
pdb.set_trace()
    data = (data / max_val) - 0.5
    x = data[:, :-1]
    y = data[:, -1]
    model = create_dense_model( x.shape[1], 1)
model.fit(x, y, batch_size=300, epochs=40,
validation_split=0.10)
model.save_weights( output_file_name )
def test_model( csv_file, model_file_name):
    data = load_simple_csv( csv_file )
max_val = np.loadtxt( 'max.txt')
    data = data / max_val - 0.5
    x = data[:, :-1]
    y = data[:, -1]
    model = create_dense_model( x.shape[1], 1)
model.load_weights( model_file_name )
    res = model.predict(x)
    y = y.reshape( res.shape )
    diff = np.abs(y - res)
    diff *= max_val[-1]
print('result ', np.mean(diff), np.std(diff), np.min(diff),
np.max(diff))
real_values = (res + 0.5) * max_val[-1]
np.savetxt( 'predict_result.txt', real_values, fmt = '%15.3f'
)
    #pdb.set_trace()
def test_rnn_model( csv_file, output_file_name ):
    data = load_simple_csv( csv_file )
max_val = np.loadtxt( 'max.txt')
    data = data / max_val - 0.5
    x = data[:, :-1]
    y = data[:, -1]
    #x_power = x[:, :-4]
    #x_temp = x[:, -4]
x_power = x[:, :-1]
x_temp = x[:, -1]
history_len = x_power.shape[1]
single_step_len = 1
x_power = x_power.reshape((-1, history_len, 1))
input_shape = (history_len, 1)
    model = create_rnn_model( input_shape, 1, 1)
model.load_weights( output_file_name )
    res = model.predict([x_power, x_temp])
y = y.reshape( res.shape )
diff = np.abs(y - res)

```

```
    diff *= max_val[-1]
print('result ', np.mean(diff), np.std(diff), np.min(diff),
      np.max(diff))
real_values = (res + 0.5) * max_val[-1]
np.savetxt( 'predict_result.txt', real_values, fmt = '%15.3f'
)
pdb.set_trace()
if __name__ == '__main__':
    csv_file = sys.argv[1]
    output_file_name = sys.argv[2]
        #train_model( csv_file, output_file_name )
        #train_rnn_model( csv_file, output_file_name )
test_model( csv_file, output_file_name )
    #test_rnn_model( csv_file, output_file_name )
```

LIST OF BIBLIOGRAPHICAL REFERENCES

- [1] Hong, T., & Shahidehpour, M. (2015). Load Forecasting Case Study. *U.S Department of Energy*. Retrieved from <https://pubs.naruc.org/pub.cfm?id=536E10A7-2354-D714-5191-A8AAFE45D626>
- [2] Sallam, A. A., & Malik, O. P. (2019b). Load Forecasting. In *Electric Distribution Systems* (pp. 41–71). IEEE. <https://doi.org/10.1002/9781119509332.ch4>
- [3] Ghalekhondabi, I., Ardjmand, E., Weckman, G. R., & Young, W. A. (2017). *An overview of energy demand forecasting methods published in 2005–2015. Energy Systems* (Vol. 8). <https://doi.org/10.1007/s12667-016-0203-y>
- [4] Singh, P., & Dwivedi, P. (2018). Integration of new evolutionary approach with artificial neural network for solving short term load forecast problem. *Applied Energy*, 217(October 2017), 537–549. <https://doi.org/10.1016/j.apenergy.2018.02.131>
- [5] Wood, A. J., Wollenberg, B. F., & Sheblé, G. B. (2013). *Power generation, operation, and control*. John Wiley & Sons.
- [6] Bharadwaj Aarti, G., & Meeta, K. (2001). Demand Forecasting for Electricity: the Indian experience. *Public Utility Research Center (PURC)*, (Environmental and Safety Issues), 175–192. Retrieved from http://regulationbodyofknowledge.org/wp-content/uploads/2013/03/Mehra_Demand_Forecasting_for.pdf
- [7] Jacobsen, H. K. (1998). Integrating the bottom-up and top-down approach to energy-economic modelling: the case of Denmark. *Energy Economics*, 20(4), 443–461. [https://doi.org/10.1016/S0140-9883\(98\)00002-4](https://doi.org/10.1016/S0140-9883(98)00002-4)
- [8] Fleiter, T., Rehfeldt, M., Herbst, A., Elsland, R., Klingler, A. L., Manz, P., & Eidelloth, S. (2018). A methodology for bottom-up modelling of energy transitions in the industry sector: The FORECAST model. *Energy Strategy Reviews*. <https://doi.org/10.1016/j.esr.2018.09.005>
- [9] Natural Resources Canada's Office of Energy. (2016). Improving Energy Performance in Canada. *Natural Resources Canada's Office of Energy Efficiency*.
- [10] Natural Resources Canada. (2018). Energy Efficiency in Canada. Retrieved from <https://www.nrcan.gc.ca/sites/www.nrcan.gc.ca/files/www/pdf/publications/emmc/parliament17-18.pdf>

- [11] Natural Resource Canada. (2016). Energy Efficiency Trends in Canada - 1990 to 2013, (March), 1–51. Retrieved from <https://www.nrcan.gc.ca/sites/www.nrcan.gc.ca/files/energy/pdf/trends2013.pdf>
- [12] Natural Resources Canada. (2014). Smart grid in Canada 2014, 40. Retrieved from https://www.nrcan.gc.ca/sites/www.nrcan.gc.ca/files/canmetenergy/files/pubs/SmartGrid_e_acc.pdf
- [13] Rahman, A., Srikumar, V., & Smith, A. D. (2018). Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks. *Applied Energy*, 212, 372–385.
- [14] Natural Resources Canada. (2017). Building Canada’s Energy Future Together. Retrieved from [https://www.nrcan.gc.ca/sites/www.nrcan.gc.ca/files/emmc/pdf/2018/en/building-canadas-energy-future-together-en-\(Clean\).pdf](https://www.nrcan.gc.ca/sites/www.nrcan.gc.ca/files/emmc/pdf/2018/en/building-canadas-energy-future-together-en-(Clean).pdf)
- [15] Dagdougui, H., Mary, N., Beraud-Sudreau, A., & Dessaint, L. (2016). Power management strategy for sizing battery system for peak load limiting in a university campus. In *2016 IEEE Smart Energy Grid Engineering (SEGE)* (pp. 308–312). IEEE.
- [16] Office of Electricity Delivery & Energy Reliability. (2016). Advanced Metering Infrastructure and Customer Systems. *Results from the Smart Grid Investment Grant Program*, 98. Retrieved from https://www.energy.gov/sites/prod/files/2016/12/f34/AMI_Summary_Report_09-26-16.pdf
- [17] BC Hydro. (2011). Smart Metering & Infrastructure Program Business Case, 44.
- [18] Prindle, W., & Koszalka, M. (2012b). Succeeding in the Smart Grid Space by Listening to Customers and Stakeholders. In *Smart Grid* (pp. 343–369). Elsevier. Retrieved from <https://www.sciencedirect.com/science/article/pii/B9780123864529000140>
- [19] Sallam, A. A., & Malik, O. M. P. (2019a). ELECTRIC DISTRIBUTION SYSTEMS. Retrieved from <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119509332>
- [20] Future, T. H. E. (n.d.). VISION 2050 THE FUTURE OF CANADA’S ELECTRICITY SYSTEM Acknowledgements. Retrieved from www.vision2050.ca
- [21] Energie et Ressources naturelles. (2016). *Energy in Quebec: The 2030 energy policy*. Retrieved from <https://mern.gouv.qc.ca/english/energy/strategy/pdf/The-2030-Energy-Policy.pdf>

- [22] Hopkins, A. S. (2017). Best Practices in Utility Demand Response Programs - With Application to Hydro-Québec's 2017–2026 Supply Plan. *Synapse Energy Economics*. Retrieved from <http://www.synapse-energy.com/sites/default/files/Utility-DR-17-010.pdf>
- [23] Hydroquebec. (2019). Hydroquebec Electricity Rates. Retrieved from <http://www.hydroquebec.com/data/documents-donnees/pdf/electricity-rates.pdf?v=20190401>
- [24] National Energy Board. (2016). Canada's Energy Future 2016 Update ENERGY SUPPLY AND DEMAND PROJECTIONS TO 2040, 1–136. Retrieved from <https://www.neb-one.gc.ca/nrg/ntgrtd/fr/2016updt/2016updt-eng.pdf>
- [25] Natural Resources Canada. (2013). Canada – A Global Leader in Renewable Energy Enhancing Collaboration on Renewable Energy Technologies. *Energy and Mines Ministers Conference*, (August), 14. <https://doi.org/10.1016/j.bbr.2010.04.014>
- [26] Hydro-Québec. (2018). Clean energy to power us all. Retrieved from <http://www.hydroquebec.com/data/documents-donnees/pdf/sustainability-report.pdf>
- [27] ISO New England. (2019). 2019 Regional Electricity Outlook. Retrieved from https://www.iso-ne.com/static-assets/documents/2019/03/2019_reo.pdf
- [28] Hong, T., Gui, M., Baran, M. E., & Willis, H. L. (2010). Modeling and forecasting hourly electric load by multiple linear regression with interactions. In *IEEE PES General Meeting* (pp. 1–8). IEEE. Retrieved from https://www.researchgate.net/publication/224179146_Modeling_and_forecasting_hourly_electric_load_by_multiple_linear_regression_with_interactions
- [29] Marino, D. L., Amarasinghe, K., & Manic, M. (2016). Building energy load forecasting using deep neural networks. In *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society* (pp. 7046–7051). IEEE. Retrieved from https://www.researchgate.net/publication/309573233_Building_Energy_Load_Forecasting_using_Deep_Neural_Networks
- [30] Wang, Z., & Srinivasan, R. S. (2017). A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models. *Renewable and Sustainable Energy Reviews*. <https://doi.org/10.1016/j.rser.2016.10.079>

- [31] Neto, A. H., & Fiorelli, F. A. S. (2008). Comparison between detailed model simulation and artificial neural network for forecasting building energy consumption. *Energy and Buildings*, 40(12), 2169–2176. Retrieved from <https://www.sciencedirect.com/science/article/pii/S0378778808001448>
- [32] Yildiz, B., Bilbao, J. I., & Sproul, A. B. (2017). A review and analysis of regression and machine learning models on commercial building electricity load forecasting. *Renewable and Sustainable Energy Reviews*. <https://doi.org/10.1016/j.rser.2017.02.023>
- [33] Armstrong, J. S. (2011a). Illusions in regression analysis. *Available at SSRN 1969740*. Retrieved from <https://www.researchgate.net/publication/228194929>
- [34] Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill, Inc., New York, NY, USA.
- [35] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436. Retrieved from <https://www.cs.toronto.edu/~hinton/absps/NatureDeepReview.pdf>
- [36] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press. Retrieved from <https://www.deeplearningbook.org/>
- [37] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Dubourg, V. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(Oct), 2825–2830.
- [38] Géron, A. (2017). *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. “O’Reilly Media, Inc.”
- [39] Krogh, A., & Hertz, J. A. (1992). A simple weight decay can improve generalization. In *Advances in neural information processing systems* (pp. 950–957). Retrieved from <https://papers.nips.cc/paper/563-a-simple-weight-decay-can-improve-generalization.pdf>
- [40] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), 1929–1958. Retrieved from <https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf>
- [41] Ripley, B. D., & Hjort, N. L. (1996). *Pattern recognition and neural networks*. Cambridge university press.
- [42] Vu Huu Tiep. (2018). *Machine Learning cơ bản*.

- [43] Demuth, H. B., Beale, M. H., De Jess, O., & Hagan, M. T. (2014). *Neural network design*. Martin Hagan. Retrieved from <https://hagan.okstate.edu/NNDesign.pdf>
- [44] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4), 303–314.
- [45] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105). Retrieved from <https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>
- [46] Nocedal, J., & Wright, S. (2006). *Numerical optimization*. Springer Science & Business Media. Retrieved from <https://link.springer.com/content/pdf/10.1007%2F978-0-387-40065-5.pdf>
- [47] Ruder, S. (2016). An overview of gradient descent optimization, 1–14.
- [48] Hagan, M. T., & Menhaj, M. B. (1994). Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6), 989–993 Retrieved from <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=329697>
- [49] Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2), 431–441. Retrieved from <http://people.duke.edu/~hpgavin/ce281/lm.pdf>
- [50] Hagan, M. T., Demuth, H. B., & Beale, M. H. (1996). *Neural network design*, PWS Pub. Co., Boston, 3632.
- [51] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533–536. Retrieved from https://www.iro.umontreal.ca/~vincentp/ift3395/lectures/backprop_old.pdf
- [52] Nielsen, M. A. (2015). *Neural networks and deep learning* (Vol. 2018). Determination press San Francisco, CA, USA:
- [53] Olah, C. (2015). Calculus on computational graphs: Backpropagation.
- [54] Hydro-Québec. (2016). Electricity Rates Effective April 1, 2016.

- [55] Dagdougui, H., Bagheri, F., Le, H., & Dessaint, L. (2019). Neural network model for short-term and very-short-term load forecasting in district buildings. *Energy and Buildings*, 203, 109408.
Retrieved from
<https://www.sciencedirect.com/science/article/pii/S0378778819304505>