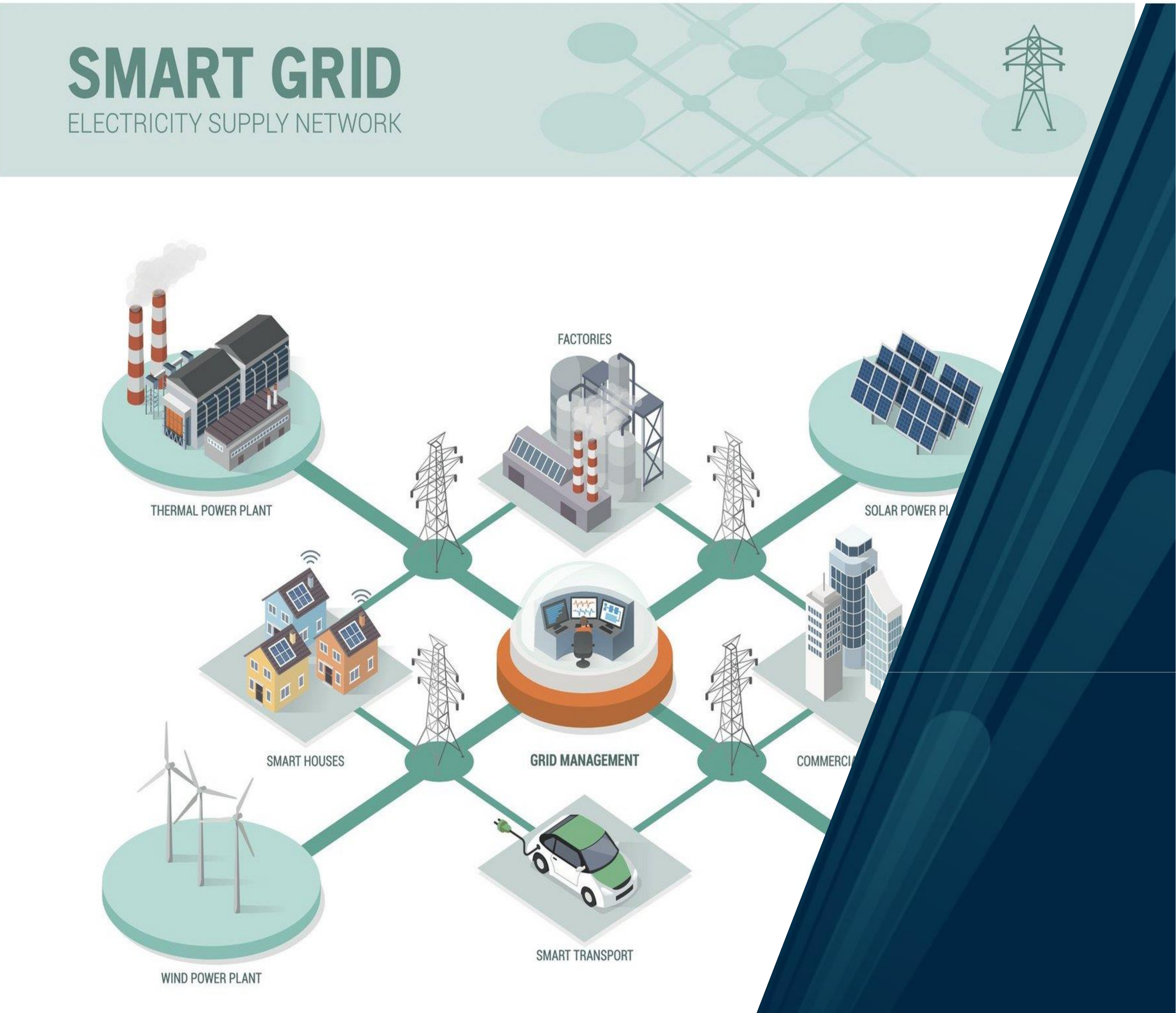
Department of Electrical Engineering

# Time Series Forecasting of Reactive Power Support from Smart Converters in SDN Using Machine Learning

Mujtaba Aziz

Candidate #: 2

Master’s thesis in Electrical Engineering…ELE-3900…May 2023



# ACKNOWLEDGEMENT

First and foremost, I would like to express my sincerest gratitude to Allah Almighty for granting me the strength, perseverance, and opportunity to complete this thesis. Without His blessings and guidance, I would not have been able to reach this point.

I would like to extend my heartfelt appreciation to my supervisors, PhD researcher Raju Wagle and Professor Pawan Sharma, for their unwavering support, guidance, and encouragement throughout my academic journey. Raju Wagle was always accessible, even during his busy schedule he took out time to guide me. His invaluable insights, constructive criticism, and expertise have greatly contributed to the success of this thesis.

I am also deeply grateful to my company manager, Christian Murrilo, for providing me with the necessary resources and support that allowed me to pursue my master's degree while working full-time. His unwavering encouragement and trust in my abilities were instrumental in keeping me motivated and focused on my goals.
Furthermore, I would like to thank my parents for their unconditional love, support, and unwavering belief in my abilities. Their encouragement and constant prayers have been a source of inspiration for me. Additionally, I would like to express my gratitude to my friends for their continuous support and encouragement throughout this journey.

Finally, I would like to extend my thanks to all those who have contributed to my academic and personal growth, directly or indirectly. Your support, guidance, and encouragement have been instrumental in shaping my personality and helping me achieve my goals.

# ABSTRACT

The climate changes in the last few years created a major need to integrate more renewable energy sources and other low-carbon technologies into the power system network. This causes to face more changes in the power system network, which are particularly visible in distribution power networks. A higher penetration of the distributed energy resources, installed at low- voltage and or medium-voltage levels, creates new challenges for distribution system operators. One of the important issues is to effectively manage the reactive power in a smart distribution network, as the mismatch of the reactive power in a power system network can cause voltage violations in the network.

By timely predicting the reactive power, a distribution system operator can make better decisions to avoid any voltage violations. Several conventional techniques like optimal power flow (OPF) control and droop control are used, which are highly dependent on grid models. But machine learning (ML) is an effective approach due to its capability of handling multiple variable data sets and its performance is being independent of grid constraints.

Therefore, in this thesis, we propose a machine learning-based approach for time series prediction of reactive power in a smart distribution network. After going through a literature review to find research gaps, a detailed methodology is discussed, highlighting tools used and how they impact on our objectives. Moving forward, a power flow analysis is performed to see the impact of reactive power on SDN. After acquiring all the data required for training algorithms, ML is implemented, and the results are also compared with the optimal power flow method. The results show that the predicted reactive power by the ML approach is very close to the OPF results. However, more improvements can be made by increasing the dataset and changing in the layers of ML algorithms.

# Table of Contents

## Contents

# List of Figures

## List of Tables

# ABBREVIATIONS

| | |
|---|---|
| OPF | Optimal power flow |
| ML | Machine Learning |
| SVM | Support vector machines |
| RES | Renewable Energy Sources |
| PV | photovoltaic |
| SDN | Smart distribution Network |
| DER | Distributed Energy Sources |
| ESU | Energy storing unit |
| CCM | Current control Mode |
| IPC | interlinking power converter |
| CNN | convolution Neural Network |
| LSTM | Long Short-term memory |
| NIDS | Network Intrusion Detection System |
| AI | Artificial Intelligence |
| MARL | Multi agent reinforcement learning |
| NAR | Nonlinear Autoregressive |
| ANN | Artificial Neural Network |
| MG | Micro Grids |
| EPF | External Penalty function |
| DNP | Deep Neural Prophet |
| GPU | Graphics Processing unit |
| IDS | integrated Distribution System |
| MPL | Multi-Layer Perceptron |

| | |
|---|---|
| FNN | feedforward neural network |
| ARIMA | autoregressive integrated moving average |
| Dec-POMDP | Decentralized Partially Observable Markov Decision Process |
| MAE | Mean Absolute Error |
| RMSE | Root mean square error |
| HV | High Voltage |
| LV | Low Voltage |
| PU | Per Unit |

# 1. Introduction

This chapter of the thesis will hover over background and motivation to work on this problem. There is an explanation of complete problem statement, a description of main objectives and scope of the thesis is discussed.

## 1.1. Background and Motivation

The population of world is growing at very fast pace and so does the energy needs. To overcome power shortage, there is a high-speed transition happening towards Renewable energy resources especially solar plants. There is huge injection of PV’s into our grid system taking place every day. These PV’s are attached with smart converters which have the ability to inject active or reactive power into our system. Since reactive power plays a very important role in mitigating voltage violations so it’s controlled injection can help us to stabilize our Distribution Networks.
There have been various conventional methods used for keeping reactive power stabilized in DN, like using OPF or Droop control but these are highly grid model dependent. With the advent of machine learning (ML), computers tend to learn from data without being explicitly programmed. By employing experience (i.e., data) to learn patterns, correlations, and insights that can be applied to new situations, machine learning aims to provide computers with the ability to automatically improve their performance at a task over time. This field has been evolving since the 1980's. It emerged as an individual field of study when researchers started to investigate different dynamics of creating machines which could learn and interact like humans. ML research expanded, with a focus on more intricate and realistic situations. The spectrum of applications for machine learning was significantly increased in the 1990s and 2000s with the invention of support vector machines (SVMs) and boosting methods [52].

ML has been researched and used in the context of power networks to solve a variety of issues, including demand-side management, grid stability, and energy efficiency. The biggest problems in implementing ML in power systems are a lack of accountability and guarantee, which cannot be tolerated given the importance of a stable power supply to the entire population. Hence, it is best to begin with less complex problems that are less sensitive to accuracy, yet difficult to address using traditional techniques.

The main motivation to work on this research problem is that it is a fusion of the power sector and the ML community. ML being a highly emerging field can be a very good tool for Electrical Engineers to sort out different problems within power industry. Hence, it can be very helpful and good start for future work as well.

## 1.2. Problem Statement

In recent times, with high penetration of PV’s in our distributed networks, the management of reactive power is becoming a challenging task. High or low injection of reactive power into the distribution network can cause voltage violations, which results in huge power losses and breakdowns. Renewable energy sources generate power that is often variable and intermittent, leading to fluctuations in grid voltage and reactive power demand. So, reactive power compensation is very important to stabilize grid voltage and ensure optimal power flow. Through the incorporation of sensors, communication systems, and sophisticated algorithms, smart distribution networks offer the opportunity for more advanced reactive power management. The creation of accurate forecasting models for RES generation, the optimization of reactive power compensation schemes, and the coordination of numerous devices and stakeholders are among the issues that still need to be resolved. To solve this problem, we can use the control feature of solar inverters to optimize voltage regulation using reactive power control. However, as there is a lot of uncertainty and variability of load, this can be a challenging task to do, especially when there is limited reactive power supply from the PV inverters. ML can be a good candidate to sort out this issue, as it allows algorithms to learn from the previous standard data without constraints of network parameters and then control can be implemented based on prediction made by ML algorithm. The efficiency of ML in regulating reactive power in a decentralized network is determined by several parameters, but most important part is the availability of a big chunk of real time data and the learning algorithm that has been chosen. As a result, before implementing the ML-based control system in a real-world application, a comprehensive analysis and assessment of system performance and resilience is required.

In this thesis we will see how the distribution network is impacted if there is high, low or no external injection of reactive power. Then we will implement a machine learning algorithm to predict time-series reactive power support from smart converters to mitigate the voltage violations in a distribution network which is our main objective.

## 1.3. Research Objectives

This thesis covers the following objectives.

- Literature survey of different machine learning approaches used in active and reactive power control.
- Analyzing the impact of reactive power support on distribution network.
- Time series forecasting of reactive power support from smart converters using the ML approach.
- Comparison of the analysis obtained from the conventional approach for forecasting with the ML based method.

## 1.4. Thesis Limitations

Here are some main limitations of this research work.

- It only offers a time-series prediction of reactive power support from smart converter. Implementing control algorithm is out of the scope of this research.
- Only 13 PV’s are used in the network for this analysis. The results may show variation when there is alteration in the number of PV penetration.
- Algorithms are trained on a very limited data set. There are high chances of performance improvement when the data set is increased because of the more pattern availability to algorithms.
- As real-time data sets were not available, so data generated by OPF analysis is used which is based on initial profile obtained by sensitivity analysis [16, 51].

## 1.5. Scope of thesis

There are several sections in this thesis which cover both theoretical and experimental aspects.

All these sections aim to cover the main objectives of this thesis.

- **Introduction**: In this section, background and motivation of thesis is discussed along with problem statement. Moreover, the main objectives and scope of this thesis is also mentioned in the first section.
- **Literature review**: This part covers a detailed overview about SDN's, smart converters, machine learning background and its applications in the field of power systems. It also discusses existing work done in the domain of predictive analysis for reactive power support.
- **Methodology**: It is one of the critical sections of this thesis and it covers the detailed process overview and tools used through which the objectives of the thesis are achieved. There is a flow diagram, as well as written details for process understanding.
- **Implementation of network and algorithms**: This section is one of the most detailed portions in this thesis and covers two main objectives of the thesis which include implementations of ML algorithms for reactive power prediction and impact of reactive power support on SDN. Aspects like mathematical modeling and theoretical knowledge are also covered in this section.
- **Results and Discussion**: This portion covers results obtained from different ML algorithms with explanations. The performance of each algorithm is analyzed in this section, and a comprehensive analysis has been done considering pros and cons of using conventional approach versus the ML algorithm for the reactive power support problem.
- **Conclusion**: This section summarizes the whole thesis and discusses the main findings. Also there is a recommendation of future work that could be done on this research.

# 2. Literature Review

This chapter will highlight some features of SDN, different types of smart converters and literature on machine learning, including some applications of machine learning in the power sector focusing on reactive power control and voltage regulation.

The growing utilization of static power electronics converters in renewable energy sources within smart distribution networks (SDNs) has led to an increasing interest in using smart converters for providing reactive power support. In this context, the use of machine learning algorithms to address uncertain system dynamics in SDNs has become a high priority within the smart grid paradigm. Therefore, the time-series forecasting of reactive power support from smart converters in SDN using machine learning has gained significant attention in recent years. This approach involves the analysis of historical data to develop predictive models that can accurately forecast reactive power support, facilitating efficient and reliable operation of SDNs.

## 2.1 Review on Smart Distribution Networks

An advanced power grid system known as a "smart distribution network" (SDN) uses digital communication and control technology to improve the efficiency, dependability, and sustainability of electricity distribution. SDNs were created to enable bidirectional power flow, real-time monitoring and management of power flows, and the integration and optimization of diverse distributed energy resources (DERs) like solar panels, wind turbines, energy storage systems, and electric vehicles [25]. In addition to enabling new business models and consumer engagement methods, the incorporation of SDN technology can enhance power quality, lower energy losses, and boost the overall flexibility and resilience of the distribution system. But in order to successfully adopt SDNs, extensive planning and coordination are necessary, as well as resolving several difficulties like cybersecurity, interoperability, and legal concerns. The research in [12] stress the necessity for more study on SDN optimization and control strategies and the significance of creating thorough and adaptable frameworks for SDN planning and management.

### 2.1.1. Features of SDN

There are many advantages of using smart distribution networks but here are presented some of the key benefits.

- **Improved monitoring and control**: SDN enables utilities to monitor and control the distribution network in real-time, allowing them to swiftly identify and address problems like voltage fluctuations, overloading, and faults that can result in power losses.
- **Predictive maintenance**: SDN can also make it possible for utilities to perform predictive maintenance by giving those insights into the state of their network's assets and spotting potential problems before they arise. This can reduce downtime and increase the network's overall dependability [4].
- **Load balancing**: By balancing loads across the network, SDN can improve power distribution and reduce the risk of overloading and other problems that could result in power losses.
- **Integration of distributed energy resources:** SDN can aid in the integration of distributed energy sources into the grid, reducing the amount of power that must be transported over long distances and, in turn, lowering power losses. Examples of such distributed energy sources are solar panels and wind turbines.
- **Minimal Impact of Fault on Customers:** Due to immediate recognition of fault location, appropriate decisions, system restoration of healthy sections, and system recovery to normal condition in the smart grid, the impact of faults on customers was minimized [5].

In summary, smart distribution networks (SDNs) are a cutting-edge response to the problems that traditional electricity distribution networks are currently experiencing. To guarantee the effective and dependable management of electricity grids, they integrate a variety of contemporary technology. Effective demand response management is made possible using SDN, which also increases the stability and resilience of the power system, lowers power losses, and enhances power quality.

## 2.2. Review on Power/Smart Converters

A smart converter, often referred to as a power electronic converter, is an equipment that changes the form in which electrical power is supplied. In electrical power systems, it is primarily used to convert DC (direct current) to AC (alternating current) or the other way around. Smart converters can control the flow of power and enhance the stability and effectiveness of power systems thanks to their sophisticated control and communication features. They are a crucial part of "smart grids," as AC and DC are synchronized in a smart grid, a multi-function converter is needed to transport electricity between the two grids [15]. The converters can function as uninterruptible power systems (UPS) to supply both DC and AC loads in the event of an AC grid failure [14].
There are several different types of power converters available today, but we will discuss some of the main types.

### 2.2.1 Grid Forming Power Converters:

Electrical devices known as grid forming power converters can provide an isolated grid or micro grid with a stable and regulated voltage and frequency, allowing it to function apart from the main grid. Grid forming power converters, in contrast to conventional grid-tied power converters, can produce a steady AC voltage and frequency on their own, enabling them to support the power system in the event of a blackout or grid outage. In isolated grids or micro grids, these converters are very helpful for incorporating renewable energy sources and energy storage systems. These converters are considered as current control voltage source converters [19, 20] because of their ability to adjust active and reactive powers, maintain the state of charge (SOC) of an energy storing unit (ESU) thus enhancing the power quality [19].

### 2.2.2 Grid Following Converters

Power electronic devices known as "grid-following converters" are developed to maintain a constant output voltage or current that follows the grid voltage or current, respectively. In order to transform the DC output of solar photovoltaic (PV) panels or wind turbines into AC power that can be synchronized with the utility grid, these converters are frequently employed in renewable energy systems, such as solar photovoltaic (PV) and wind power systems. Grid following converters can operate in current control mode (CCM) or power control mode [28]. These modes achieve active and reactive power control (for AC sub-grid) and current or power control (for DC sub-grid).

Grid-following converters have the benefit of operating in parallel with the grid without the requirement of synchronization equipment and by supporting the grid with reactive power [19]. In summary, grid forming power converters are essential components in current power systems because they offer grid support services that promote the stability and dependability of the grid.

### 2.2.3 Interlinking Power Converters:

In order to create the AC/DC MG, the interlinking power converter (IPC) is typically employed to connect the DC and AC sub-grids. Based on the power surpluses on a specific sub-grid, these converters can operate in both directions. It has three different operating modes: inversion, rectification, and stop mode. The IPCs can connect to many grids as a result of recent improvements in power converters. Numerous studies have suggested that IPCs can potentially serve as power converters that construct and support grids [33]. However, the literature [29] highlighted some serious problems, including 1) non-linear load behavior when transferring power from an AC sub-grid to a DC sub-grid, 2) current circulation between parallel-operated IPCs, and 3) re-synchronization problems following the removal of faults. Additionally, by restricting the power exchange across sub-grids, the usage of IPCs for ancillary services like power quality enhancement may lower the maximum RES utilization, which lowers system efficiency. As a result, the effectiveness of system and conversion quality must be traded off.

In general, power systems need smart converters for several reasons. First, they make it possible to integrate renewable energy sources such as solar and wind power into the grid in a reliable and efficient manner. Second, by controlling voltage and frequency, lowering harmonic distortion, and providing reactive power assistance, they also improve power quality. Third, they improve grid resilience and stability by minimizing grid disturbances and providing quick and precise control over power flow. Fourth, they make it possible to monitor and communicate in an advanced manner, which improves the management and coordination of dispersed energy resources.

## 2.3 Introduction to Machine Learning

A growing requirement for decision-support technologies is a direct result of the explosion in data available over the past few decades. Machine learning is one such technology; it is a branch of AI that allows machines to learn from experience and examples without being explicitly programmed. This involves feeding data to an algorithm to build a model, which can then be used to make predictions or uncover patterns. The fields of healthcare, finance, media industry, image and video processing [22], are just a few examples of the numerous fields that can benefit from machine learning. This article provides an overview of the basics of machine learning, its algorithms, and its uses in various fields [1].

Supervised type of learning [22] involves having both input and output variables, and an algorithm can derive a function that maps the input to the output. It is used frequently when data is already available for the target output [42]. Separate sets of data, known as training and test data, are used by the algorithm to learn a predictive or classifying function.

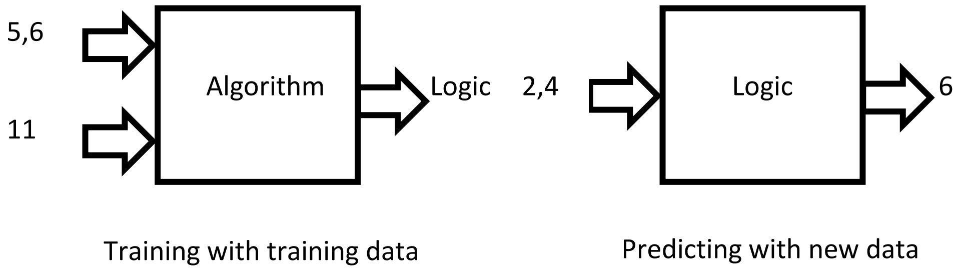


*Figure 1 Example of Supervised Learning*

Unsupervised form of learning [22] involves only having input variables, with unlabeled data that is utilized by algorithms to identify patterns of interest. The algorithms learn the features independently, and when new data is presented, they use the previously learned features to group the data accordingly [27]. It is like having questions without correct answers to match them with. This type of learning is used primarily for clustering and association problems [2].

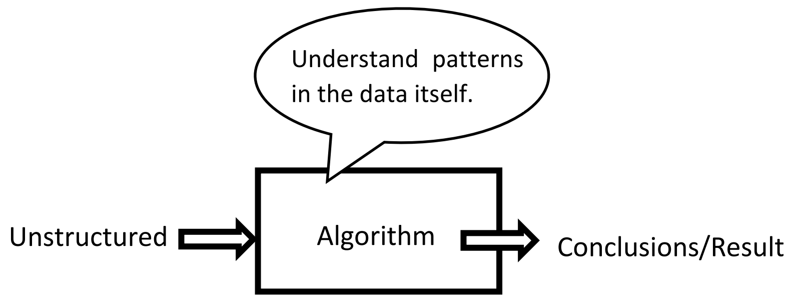


*Figure 2 Unsupervised learning model*

### 2.3.1. Machine Learning Approaches

In the realm of power systems, reactive power control is a critical task that ensures power quality and system stability. With the rise of smart distribution networks, several machine learning approaches can be employed for reactive power control. These approaches include Deep Learning, support vector machines, Decision Trees, Random Forests, and Gradient Boosting methods [27].

Artificial Neural Networks

Support Vector Machines

Decision Trees

Random Forest Algorithm

Deep Learning Algorithms

Gradient Boosting

#### Artificial Neural Networks

By analyzing past data and taking into account other aspects of the system, artificial neural networks (ANNs) can be put to use in reactive power control. Artificial neural networks (ANNs) are a class of machine learning algorithms that, through training, can discover previously unseen patterns and relationships in data [34]. It is up to the designer to decide on the ANN's architecture, which includes the number of input nodes, hidden layers, and output nodes. The level of detail and complexity of the system might inform the design [27]. By changing the weights and biases of the nodes, training the ANN using historical data allows for the error between predicted and actual reactive power values to be minimized.

**Support Vector Machines**

Support Vector Machines (SVMs) can be used in reactive power control to predict the reactive power requirements based on historical data and system parameters. For both classification and regression, SVMs are a useful machine learning technique. [17]. Designing the SVM model entails choosing the kernel function and fine-tuning the hyper-parameters. The decision boundary's form is determined by the kernel function, which can be linear, polynomial, or a radial basis function (RBF) [17]. After determining the best hyperplane for categorizing the reactive power data, the SVM is then trained using the historical data. Good empirical results and strong theoretical foundations are offered by support vector machines. Applications include text classification, object recognition, and recognition of handwritten digits [7].

**Decision Trees**

Decision trees can be used in reactive power control forecasting to analyze power system data and build a model that predicts the reactive power requirements [37]. The decision tree algorithm segments information into subsets determined by the input variables values, modeling choices and their outcomes in a tree structure [27, 37].  Steps for Implementing DT’s in reactive power control are almost similar to those support vector machines although computational power of both algorithms are different.

**Random Forest**

Random Forest can be used to forecast reactive power demand based on historical data. The algorithm functions by first constructing several decision trees, each based on a unique subset of the training data, and then averaging the predictions from these trees. This helps to increase prediction accuracy by lowering the likelihood of overfitting [27].

To use Random Forest for the forecasting of reactive power support, historical data on reactive power demand, weather conditions, and other relevant factors can be collected. The reactive power demand can be predicted by using these data sets to train a Random Forest model. Reactive power

generation and consumption can be optimized with this knowledge, leading to greater system stability [37].
Overall, Random Forest is a useful tool for reactive power forecasting in power systems. It can help to improve the accuracy of predictions and optimize the control of reactive power generation and consumption for better system stability.

**Gradient Boosting**

In reactive power control forecasting, Gradient Boosting can be used to predict reactive power demand based on historical data, weather conditions, and other relevant factors. The algorithm works by iteratively adding weak models to the ensemble, and each model attempting to correct the errors of the previous models. This process continues until the ensemble can accurately predict the reactive power demand [38].

Compared to Random Forest, Gradient Boosting can be more computationally intensive and requires more tuning of hyper-parameters. However, it often achieves higher prediction accuracy and is more resistant to overfitting.

To use Gradient Boosting for reactive power control forecasting, historical data on reactive power demand, weather conditions, and other relevant factors can be collected. The Gradient boosting model, which may predict future reactive power consumption, can be trained using these data sets. Similar to Random Forest, Gradient Boosting can be used to determine which attributes have the greatest impact on reactive power consumption. System stability can be enhanced by employing this data to fine-tune the regulation of reactive power generation and consumption [38].
In summary, Gradient Boosting is an effective method for anticipating reactive power regulation in power systems. It can help to optimize reactive power generation and consumption, leading to greater system stability and improved prediction accuracy.

## Deep Learning

Reactive power forecasting is an important problem in the operation and planning of power systems. Deep learning methods have been successfully applied to this problem and have shown promising results.

This task is often accomplished by employing a recurrent neural network (RNN), such as a Long Short-Term Memory (LSTM) network. These networks are particularly useful for time-series forecasting problems, and have been used to represent the dynamic behavior of power systems. LSTM networks can learn long-term dependencies and capture complex patterns in the input data [39].

To achieve the same end of feature extraction from raw data, convolutional neural networks (CNNs) are another viable option. By representing the series as a 1-dimensional image, convolutional neural networks (CNNs) can be applied to time series data, just as they have been successful in image and speech recognition applications. A fully connected neural network (FCNN) can be trained with the CNN's output to make more accurate predictions [40].

Reactive power forecasting has also been explored with other deep learning techniques, such as deep belief networks (DBNs) and auto encoders. The reactive power generated by wind turbines has been accurately predicted using DBNs because of their ability to learn hierarchical representations of the input data. In order to lessen the burden on the computer's processing power, auto encoders can be utilized for feature extraction and dimensionality reduction [39].

## Types of Methods in Deep Learning

### Recurrent Neural Networks (RNNs)

Recurrent neural networks (RNNs) are a type of neural network that excels at time series forecasting. The Long Short-Term Memory (LSTM) network is a common type of RNN that has found widespread application in time series forecasting. Long short-term memory (LSTM) networks are able to learn long-term dependencies and can effectively capture complicated patterns in the incoming data. By feeding historical data into the network, the LSTM can predict future reactive power output based on past behavior [41].

**Convolutional Neural Networks**

Although CNNs are most often employed for image identification applications, they can also be used for time series data. When making predictions using reactive power, it is helpful to think of the input data as a 1-dimensional image, with the time series serving as the horizontal axis and the input variables providing the vertical axis. For more precise predictions, one can use the CNN's output to train a fully connected neural network [42].

**Deep Belief Networks**

Unlike traditional neural networks, DBNs are capable of learning tree-like representations of their inputs. They excel in reactive power forecasting and other tasks where high-dimensional, noisy input data is present. One study found encouraging results when using DBNs to predict wind turbine reactive power output [39].

**Auto encoders**

When it comes to feature extraction and dimensionality reduction, neural networks such as auto encoders come in handy. In order to function, they reduce the dimensionality of the incoming data and then use this representation to recreate the original data. This can be helpful in finding the most crucial input variables and minimizing the computational complexity of the forecasting model [43].

## 2.4 Applications of Machine Learning in power systems

There are several different areas in power systems where machine learning is very useful and currently making quite an impact. Some of those are discussed below with a focus on reactive power control.

### 2.4.1. Power System Protection Based on Machine Learning

To overcome the difficulty of validating deep learning methods for power system protection, a new adaptive protection approach based on probabilistic machine learning is being used, specifically Gaussian discriminant analysis, for inverse-time relays. This method uses power system data to generate operation curves that can be used to verify and validate protection decisions. It demonstrates the effectiveness of the approach for fault detection, localization, relay coordination, optimization, and protection management by specifying minimal communication requirements for adapting relays. It applies the method to the CIGRE medium voltage grid to show its effectiveness under different operating conditions, including islanded grid-connected modes, radial and mesh topology, and distributed generation [44].

### 2.4.2. Networks Intrusion Detection Systems

Data transfer via networks has increased as technology and the Internet have spread. This is unfortunate since it raises the possibility of malicious assaults leading to security breaches and data corruption. Many intrusion detection technologies, such as Network Intrusion Detection technologies (NIDS), have been created to counteract this. It is common practice to apply machine learning and deep learning algorithms for developing secure NIDS. The deep learning algorithms deliver top-notch precision and efficiency [39]. The use of machine learning and deep learning algorithms, focus on the KDD Cup'99 dataset. It evaluates the accuracy of various methods and identifies problems and obstacles in NIDS [8].

As data becomes more widespread and accessible, cybersecurity has become increasingly important. Hackers and intruders are always looking to infiltrate the data, which makes network intrusion detection systems crucial. Different techniques for network intrusion detection systems highlight the importance of feature selection in machine learning algorithms. Machine learning algorithms require an additional process for feature selection, whereas deep learning algorithms include feature selection in their layers [45]. This is why deep learning algorithms tend to outperform machine learning algorithms. It concludes that deep learning tools are becoming the trend in the building of NIDS. However, deep learning requires more computations and GPUs, and there is still room for improvement in real-time detection. The literature review shows that deep learning methods are more accurate than machine learning methods, with better runtime complexity, but efficiency for real-time detection still needs improvement [8].

### 2.4.3. Voltage/VAR Control Strategy

A technique used in power systems to control and sustain voltage and reactive power flow in the transmission and distribution networks is called voltage and VAR (Volt-Ampere Reactive) control strategy. To keep the voltage within reasonable bounds and reduce power losses, it entails adjusting the magnitude of the voltage and injecting reactive power at various locations in the system. Volt-VAR control can be done in a variety of ways, including manually, time-based, and automatically using cutting-edge technologies like artificial intelligence (AI) [9] and machine learning (ML). Artificial intelligence (AI) techniques have intelligence features that traditional methods lack. As a result, reactive voltage control applications of AI techniques have received considerable attention, and several authors have found significant success in this area. The advancement of new automatic intelligent Volt/VAR control systems made possible by artificial intelligence (AI) technology will help to reduce the negative effects of the human factor and the inadequacies of conventional control systems [11, 51]. These methods can help the power system run more efficiently, cut energy costs, boost the reliability and stability of the grid, and increase the quality of the power.

### 2.4.4. MARL for Active Voltage Control

Due to the growing prevalence of de-carbonization, this research explores the application of multi-agent reinforcement learning (MARL) to address the challenge of active voltage management in power distribution networks. In this research, we present a Dec-POMDP (Decentralized Partially Observable Markov Decision Process) framework for modeling decision-making in multi-agent systems, and apply it to the active voltage control problem [10].

To facilitate collaboration between the power and MARL communities, the paper presents an open-source environment that can be used to simulate power distribution networks and test different control strategies. This environment provides a common platform for researchers to evaluate their approaches and share their results, which can accelerate progress in this area.

The paper also examines the challenges of using MARL for this problem, such as interpretability. Since MARL involves multiple agents learning and interacting with each other, it can be difficult to understand how each agent is making decisions and what factors are influencing its behavior. The paper suggests that interpretability techniques such as attention mechanisms and visualization tools can help address this challenge.

Finally, the research compares the performance of MARL to traditional control methods for active voltage control. The results show that MARL can achieve better performance than traditional methods, especially in complex scenarios with multiple agents and changing conditions [23, 31].

### 2.4.5. Forecasting of Reactive Power Consumption using ANN

Power factor polarization in individual phases is a problem for many energy consumers, especially small and medium-sized businesses. Because of the inductive and capacitive power factor variations between phases, conventional 3-phase reactive power compensators are difficult to install. Therefore, consumers pay more when their power factors exceed than those specified in their contracts. Reactive power forecasting using artificial neural networks is one solution to this problem. The research used a Nonlinear Autoregressive (NAR) neural network to forecast reactive power generation, and studied it with different input vector sizes and hidden layer neuron counts. The simulation results were compared to actual measurements, and the findings suggest that it is possible to forecast the trend of reactive power, allowing optimal planning of reactive power compensation strategies [24].

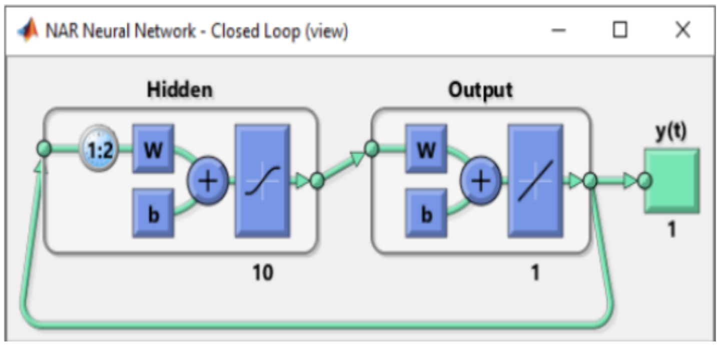


*Figure 3 Two Layer Neural Network in MATLAB [24].*

### 2.4.6. Droop Control Using Machine Learning

Using machine learning techniques like artificial neural networks (ANN) and fuzzy logic, this article discusses a droop control strategy for inverter-based micro-grids. In terms of frequency regulation and load sharing, the proposed method is found to perform better than conventional droop control methods. Artificial neural network-based controllers are used to build a droop control strategy for grid-connected inverters. Training data from simulations and experimental results are used to refine the proposed controller. The results show that the proposed controller performs better than traditional droop control methods in terms of transient response and steady-state performance [20, 21].

### 2.4.6.1 Application of Droop Control

Active and reactive power sharing across numerous dispersed generators can be accomplished with droop control. Systems with high inductive line impedances can benefit from the conventional droop approach, whereas systems with high resistive line impedances can benefit from the inverse droop method. The inverse droop control is commonly employed in primary controllers of low-voltage micro grids with predominantly resistive line parameters to accomplish decentralized power sharing. Because of the high resistance of the network, the inverse droop control has been used in many low-voltage micro-grid studies.

The suggested technique estimates the voltage and frequency trajectories of the next time step using droop constants and ideally predicted powers. This method's key benefit is that it is simple to apply and does not necessitate fine-tuning any parameters in the supplementary control layer. In addition, it does not require complex estimating methods or time-consuming model calculations. It is important to note that PI controllers are not used in the hierarchical control layers that are the focus of this method. Both theoretical analysis and empirical data show that this approach is superior to PI-based controllers in terms of response and efficiency. It has been demonstrated that the controller is reliable, has fast dynamic feedback, and can successfully reestablish voltage stability [21].

## 2.5. Concluding Literature Review

As of today, grids have Integrated Distribution Systems (IDS) which refer to the combination of various energy sources, loads, and distribution systems into a unified power grid. Reactive power control is an essential aspect of IDS management, as it helps to maintain grid stability and reliability by regulating voltage levels. Machine Learning (ML) algorithms can be used to enhance the effectiveness of reactive power control in IDS.

After reviewing the literature on this topic, it can be concluded that in terms of active power control there is ample amount of work already done in that domain, but reactive power control is not that much explored field. There is huge potential to work on this topic. Efficient algorithms like CNN, LSTM and Neural Prophet are still open to be used. The architecture of these algorithms can produce highly reliable results for time series prediction of reactive power. By analyzing real-time data and developing optimized control strategies, ML-based systems can enhance grid stability and reliability, reduce energy losses, and minimize system downtime.

# 3. Methodology

This chapter will cover the process and the tools which were used to achieve the objectives.

## 3.1 Flow Chart

Figure 4: Methodology Flowchart depicts the methodology used to develop a model for reactive power prediction in a power system using Machine Learning (ML) as the main tool. The methodology starts with identifying the problem, reviewing the related literature, moving forward towards the preparation of suitable tools for implementation of main components for analysis, including CIGRE Network topology, Power Flow analysis, and Optimal Power flow analysis. Finally, the methodology involves the analysis of the impact caused by reactive power on SDN, data generation for ML, training and testing of the ML models, and generation of results.

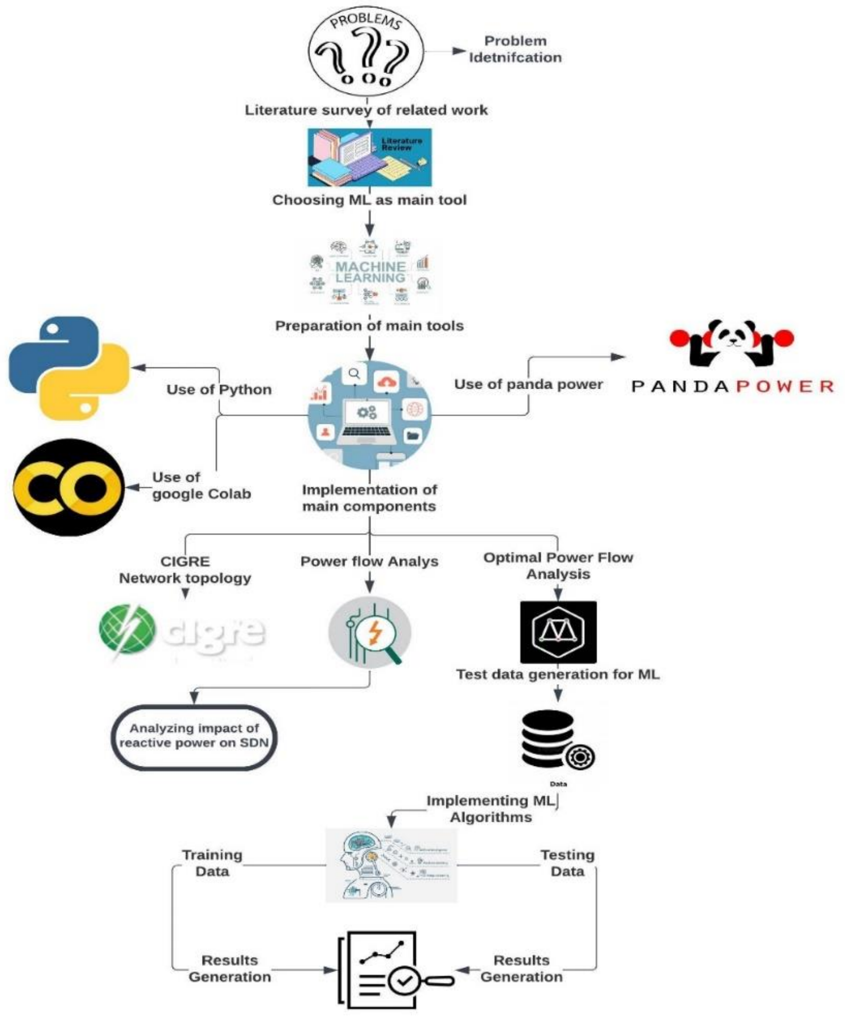


*Figure 4: Methodology Flowchart*

## 3.2 Details about Methodology

### 3.2.1. Problem Statement Identification

The first step of the methodology is to identify the problem statement, which involves defining the research question and the objectives. This step is crucial for setting the direction of the research and ensuring that the research addresses a relevant problem.

### 3.2.2. Review of Literature

After identifying the problem statement, the next step is to review the literature related to the particular use case. This step involves identifying the key theories, concepts, and research gaps related to the problem statement and determining their relevance to it.

### 3.2.3. Tool Preparation

Once the literature review is complete, the methodology progresses to the preparation of suitable tools for implementation. The tools selected for this study include Panda Power, Jupyter Notebook and Google Colab for ML. These tools are essential for implementing the main components of the analysis.

### 3.2.4. Implementation of Main Components

The next step of the methodology is the implementation of the main components for analysis, which includes CIGRE network topology, power flow analysis, and optimal power flow analysis (Mathematical Modeling and objective function development). These components are crucial for the analysis of the power system and the development of the reactive power prediction model.

### 3.2.5. Analysis of Reactive Power Impact on SDN

This step-in methodology involves the analysis of the impact of reactive power on SDN. Power flow analysis is used as a tool which helps to determine voltage profile and loadings of each bus giving the changes in the SDN due to reactive power flow and its impact on the performance of the power system.

### 3.2.6. Data Generation for ML Algorithm

The next step is to generate data for the ML algorithm. Optimal power flow analysis is done to obtain a standardized set of data on which training of algorithms is possible. The OPF gives estimated reactive power needs to be injected into the system from each DER in order to mitigate voltage violations in the system. Hence, that data is used for training, testing and comparison purposes.

### 3.2.7. Training and Testing of Model

After data generation is complete, the methodology progresses to the training and testing of the models. This step involves using the ML algorithms to train the models and evaluate their performances through testing.

### 3.2.8. Results Generation

Finally, the methodology involves the generation of results. This step includes analyzing the performance of the reactive power prediction model and presenting the findings in a clear and concise manner.

## 3.3 Tools Used

To achieve best results and match with standard work that is being used by the research community it is very important to choose the tools wisely to conduct research. In this research some standard tools have been used and this part will depict the main aspects of those.

### 3.3.1. CIGRE Network

The development of CIGRE-Networks by the CIGRE Task Force C6.04.02 [36] provides a platform for stakeholders in the energy sector to collaborate and share knowledge and experiences, leading to a better understanding of the challenges and opportunities presented by DER integration. Our work utilizes CIGRE networks to support the analysis and verification of novel methodologies and techniques aimed at promoting the economic, resilient, and eco-friendly integration of DER. This section discusses the various aspects of CIGRE networks that make them an ideal choice for our work.

CIGRE Network is chosen in our work because of the following aspects [52]:

- **Platform for Collaboration**: CIGRE networks provide a platform for stakeholders in the energy sector to collaborate and share knowledge and experiences. This is particularly important for promoting the integration of DER, which requires the input of multiple parties, including utilities, regulators, and technology providers.
- **Expertise**: CIGRE networks are developed by experts from a wide range of fields within the energy sector. This expertise is invaluable for supporting the analysis and verification of novel methodologies and techniques aimed at promoting the economic, resilient, and eco-friendly integration of DER.
- **Best Practices**: CIGRE networks develop and promote best practices for the integration of DER. This ensures that our work is based on the latest industry standards and guidelines and is therefore more likely to be widely accepted and adopted.
- **International Perspective**: CIGRE networks have a global reach, which provides an international perspective on the integration of DER. This is particularly important given the increasingly interconnected nature of the energy sector and the global nature of the DER market.
- **Focus on Research and Development**: CIGRE networks have a strong focus on research and development, which aligns well with our goals of promoting the economic, resilient, and eco-friendly integration of DER. This ensures that our work is based on the latest research and cutting-edge technologies in the field.

In this work we have used a medium voltage network with all DER which is a type of CIGRE network. This standardized network contains additional 15 distributed energy resources compared to medium voltage distribution network [36].

- 8 photovoltaic generators
- 1 wind turbine
- 2 Batteries
- 2 residential fuel cells

- 1 CHP diesel
- 1 CHP fuel cell

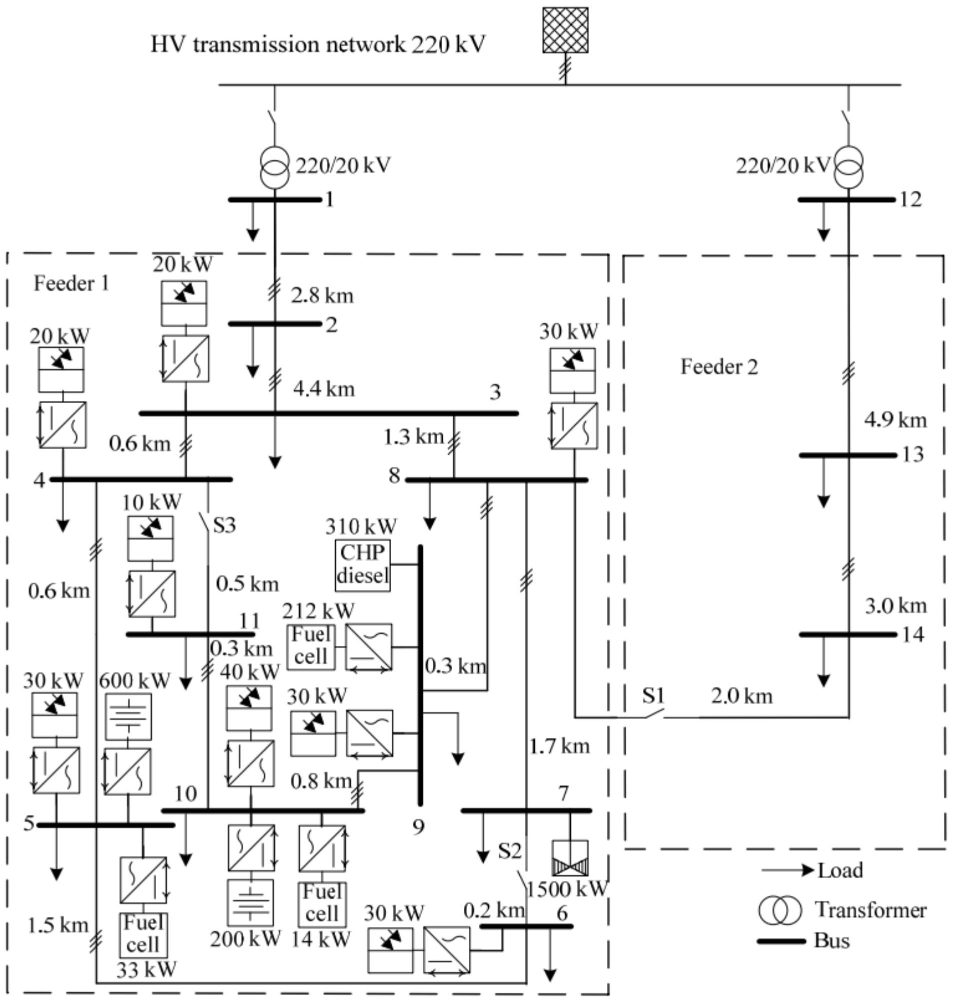


*Figure 5 CIGRE Medium Voltage Distribution Network [36]*

Figure 5 CIGRE Medium Voltage Distribution Network shows a network diagram of the CIGRE medium-voltage distribution network with distributed energy resources (DER) integrated. The nodes represent different components of the network and the arrows represent the flow of electricity. The thickness of the arrows represents the magnitude of the electricity flow, and the colors represent different voltage levels. This shows the behavior of the network with DER integration and helped us for planning and optimizing the network.

In our work, we have used the modified CIGRE network with DER integration as a basis for simulating and analyzing the behavior of the network under different scenarios. We have utilized the modified network to test and verify novel methodologies and techniques aimed at promoting the economic, resilient, and eco-friendly integration of DER.

The CIGRE network diagram provides a realistic representation of a medium voltage distribution network with DER integration. It allows us to simulate the behavior of the network under different scenarios, such as changes in load demand or changes in the availability of renewable energy sources. This helps us to evaluate the effectiveness of our proposed methodologies and techniques in promoting the economic, resilient, and eco-friendly integration of DER [35].

### 3.3.2 Google Colab & Visual Studio:

Google Colab and Visual Studio are two popular tools used in machine learning development. Although they serve different purposes and offer unique benefits, they have both been utilized in this thesis to take advantage of their respective strengths.

Google Colab is a cloud-based Jupyter notebook environment that allows users to execute Python code, create and share documents, and incorporate text, code, and visualizations [52]. Its popularity among machine learning professionals is due to several advantages, such as:

- **Free access to resources**: Colab provides free use of high performance processing tools such as GPUs and TPUs, which are essential for deep learning model training [13]. This saves money compared to building a machine learning system, as no actual hardware is required.
- **Sharing and collaboration**: Colab enables groups to work together on machine learning tasks by allowing multiple users to access the same notebook simultaneously.
- **Pre-installed libraries**: Colab comes with pre-installed libraries like TensorFlow and PyTorch, making it simple to get started with machine learning [13].

On the other hand, Visual Studio is an integrated development environment (IDE) that provides various tools for software creation, including an add-on for machine learning named Visual Studio Tools for AI [6]. This tool offers advantages such as:

- **Debugging and benchmarking tools**: These tools can be used to analyze and improve ML models.
- **Refactoring and code completion**: Visual Studio has robust features for code completion and refactoring that can help make your code more effective and efficient.
- **Azure integration**: Visual Studio seamlessly integrates with Microsoft Azure, making it easy to manage and execute different machine learning models on the cloud [6].

In conclusion, Visual Studio is better suited for more complex machine learning projects that require advanced debugging and optimization tools, while Google Colab is ideal for quick prototyping and experimentation. Therefore, in this thesis, we have used both tools to take advantage of their unique aspects and benefits.

### 3.3.3 Panda Power

Panda Power has been used in our work for power flow analysis because it is a versatile and customizable tool that provides flexibility in power system modeling and analysis. It allowed quick modification in the power flow algorithms to suit various power system models and situations, which is essential in analyzing the economic, resilient, and eco-friendly integration of DER. Moreover, Panda Power is easy to integrate with other well-known Python tools, making data analysis and visualization for power systems simpler [46].

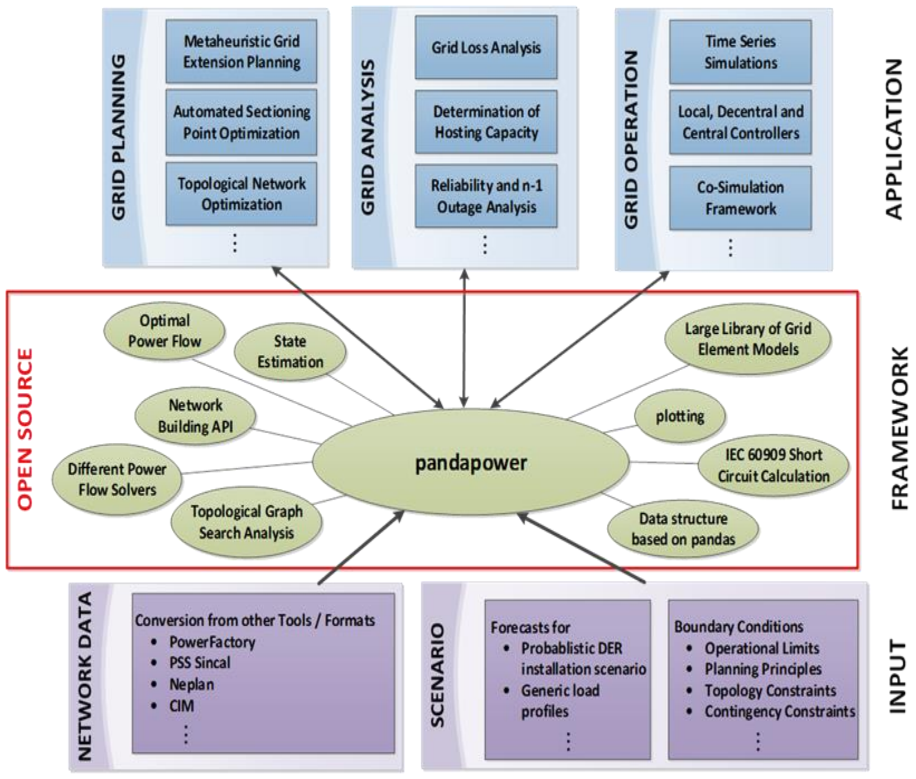

GRID PLANNING
Metaheuristic Grid Extension Planning
Automated Sectioning Point Optimization
Topological Network Optimization
GRID ANALYSIS
Grid Loss Analysis
Determination of Hosting Capacity
Reliability and n-1 Outage Analysis
GRID OPERATION
Time Series Simulations
Local, Decentral and Central Controllers
Co-Simulation Framework
APPLICATION
OPEN SOURCE
Optimal Power Flow
State Estimation
Network Building API
Different Power Flow Solvers
Topological Graph Search Analysis
pandapower
Large Library of Grid Element Models
plotting
IEC 60909 Short Circuit Calculation
Data structure based on pandas
FRAMEWORK
NETWORK DATA
Conversion from other Tools / Formats
PowerFactory
PSS Sincal
Neplan
CIM
SCENARIO
Forecasts for
Probablistic DER installation scenario
Generic load profiles
Boundary Conditions
Operational Limits
Planning Principles
Topology Constraints
Contingency Constraints
INPUT

*Figure 6 Panda Power Features [46]*

The Figure 6 Panda Power Features depicts the different features of Panda Power and how it can be useful in planning and analyzing power systems. By using Panda Power, we were able to customize and adapt our power flow algorithms to suit various power system models and situations, enabling us to perform power flow simulations more efficiently.

# 4. Implementation of Networks and Algorithms

After selecting the appropriate tools for conducting research and experiments, the main portion of the thesis focuses on the implementation of networks, power flow analysis, and machine learning (ML) algorithms. In this section, we will cover the following objectives:

1. Investigating the impact of reactive power support on smart distribution networks.
2. Collection of data using the Optimal Power Flow (OPF) method.
3. Time series prediction of reactive power support using CNN, Neural Prophet and Long Short-Term Memory (LSTM) models.

## 4.1 CIGRE Network

In order to conduct comprehensive studies on power flows, it is essential to implement an appropriate network topology. In this thesis, we utilized a medium voltage distribution network with all distributed energy resources (DER) which is a type of CIGRE network. This choice of network topology allowed us to explore and analyze the impacts of reactive power support on smart distribution networks in a realistic and practical setting.

We used Panda Power to implement the CIGRE Medium voltage network topology, which consists of 14 buses. The **benchmark topology was modified** to consider different types of cases while testing power flows. The Figure 7 Modified Layout of CIGRE MV Network with Loads and DER's [16, 30] below shows the topology of the CIGRE network we used in our thesis.

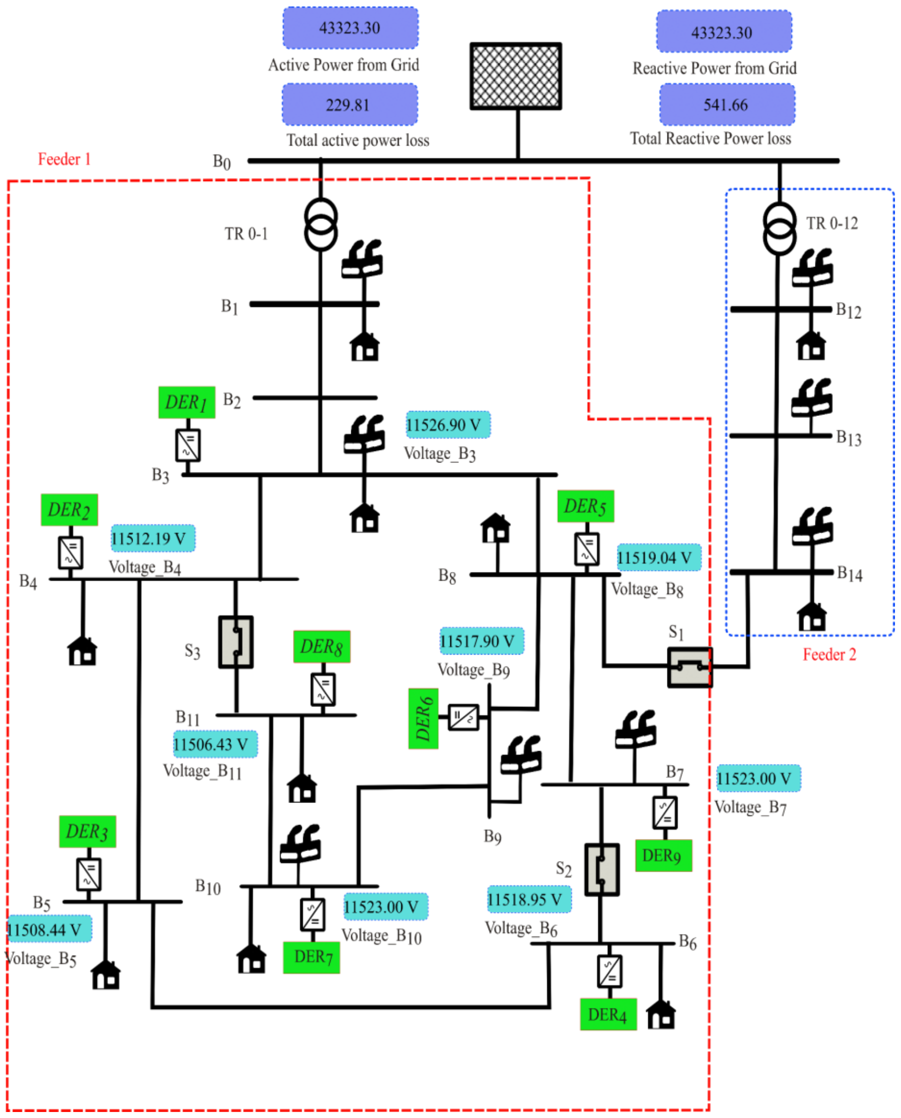


*Figure 7 Modified Layout of CIGRE MV Network with Loads and DER's [16, 30]*

The CIGRE Medium Voltage distribution network that we have implemented in this thesis consists of a total of 14 buses. Within this network, there are 9 DER's (Distributed Energy Resources), 10 residential loads, 8 industrial loads, and 2 transformers. These components have been carefully chosen to represent a real-world scenario of a smart distribution network with a mix of different load types and distributed energy resources [16].

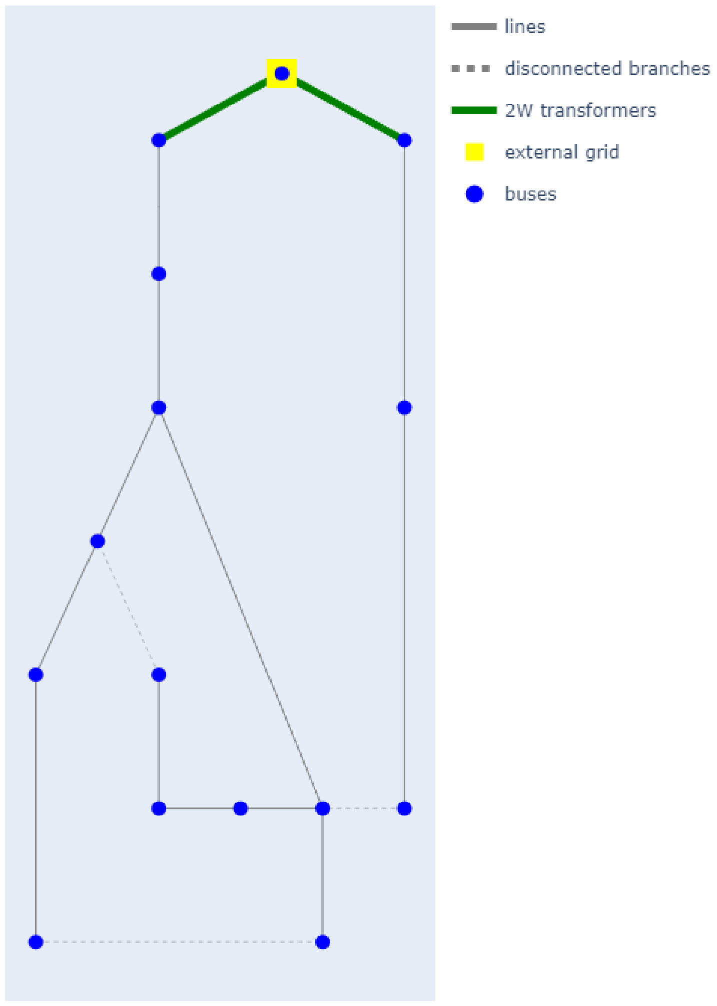


*Figure 8 CIGRE Implementation in Panda Power*

For the implementation of the CIGRE network in Panda Power, the network structure and parameters of each load and bus were added. Figure 8 CIGRE Implementation in Panda Power depicts the structure of the network after implementation, including the modified parameters. Specifically, some parameters were changed, which are detailed below:

**Transformer parameters:**

Transformer parameters were adjusted according to our problem and the table below is attached which clearly shows the modified parameters.

*Table 1 Modified Transformer Parameters [35]*

| | **Name** | **Std_ type** | **HV _ bus** | **LV _ bus** | **Sn_ MVA** | **Vn_ Hv_kv** | **Vn_ Lv_kv** | **Vk_ %** | **Vkr %** | **Pfe_ kw** | **I0 %** | **Tap_ side** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Trafo 0-1 | None | 0 | 1 | 25.0 | 110.0 | 20.0 | 12.0010 | 0.41 | 14 | 0.07 | HV |
| 1 | Trafo 0-12 | None | 1 | 12 | 25.0 | 110.0 | 20.0 | 12.0010 | 0.41 | 14 | 0.07 | HV |

Here in Table 1 Modified Transformer Parameters [35], we can observe that Tapping was introduced on the HV side, and the position of the transformer was adjusted at bus 1 and bus 12, as compared to the benchmark parameters. These standards are taken from study in paper [35].

**Load Parameters:**

Load parameters were modified to observe the impact of under and over injection of reactive power from DERs.

*Table 2 Modified Load Parameters [51, 30]*

| | **Name** | **Bus** | **P_MW** | **Q_MVAR** | **Sn_MVA** | **Scaling** | **In Service** | **Type** |
|---|---|---|---|---|---|---|---|---|
| 0 | Load R1 | 1 | 19.83900 | 4.637136 | 20.400 | 1.0 | True | Wye |
| 1 | Load R3 | 2 | 0.50170 | 0.208882 | 0.550 | 1.0 | True | Wye |
| 2 | Load R4 | 3 | 0.43165 | 0.108182 | 0.445 | 1.0 | True | Wye |
| 3 | Load R5 | 4 | 0.72750 | 0.182329 | 0.750 | 1.0 | True | Wye |
| 4 | Load R6 | 5 | 0.54805 | 0.137354 | 0.565 | 1.0 | True | Wye |
| 5 | Load R8 | 6 | 0.58685 | 0.147078 | 0.605 | 1.0 | True | Wye |
| 6 | Load R10 | 7 | 0.54330 | 0.161264 | 0.570 | 1.0 | True | Wye |
| 7 | Load R11 | 8 | 0.32980 | 0.082656 | 0.340 | 1.0 | True | Wye |
| 8 | Load R12 | 9 | 20.0100 | 4.693341 | 20.580 | 1.0 | True | Wye |
| 9 | Load R14 | 10 | 0.54005 | 0.257713 | 0.605 | 1.0 | True | Wye |
| 12 | Load C17 | 11 | 0.07650 | 0.047410 | 0.090 | 1.0 | True | Wye |
| 13 | Load Cl9 | 12 | 0.57375 | 0.355578 | 0.675 | 1.0 | True | Wye |
| 14 | Load Cl13 | 13 | 0.03400 | 0.021071 | 0.040 | 1.0 | True | Wye |

In Table 2 Modified Load Parameters [51, 30], residential loads are represented by loads with R series, while commercial loads are represented by loads with C series. In industries, both resistive and inductive loads are common, making this configuration suitable for a wide range of scenarios. The load parameters are presented in Table 2 Modified Load Parameters [51, 30], which lists the values of active power (P), reactive power (Q), and voltage (V) for each load.

## 4.2. Impact of Reactive Power Support on SDN

This part covers one of the main objectives of our thesis which is to analyze the impact of reactive power support on SDN and implementation of CIGRE Network was prerequisite to proceed with this.

### 4.2.1 Power Flow analysis

An electrical power system's power flow is investigated using a technique called load flow analysis, also referred to as power flow analysis [52]. A group of nonlinear equations known as the power flow equations are used to model the system's power movement.
Complex power, of which active power is the real portion and reactive power the imaginary part, can be used to express the power flow equations. Considering a system with “n” buses. For each bus i, the power flow formulae can be expressed as follows:

$$P_i = V_i \sum \quad (G_{ij} V_j cos(\Theta_i - \Theta_j) + B_{ij} V_j sin(\Theta_i - \Theta_j))$$

$$Q_i = V_i \sum \quad (G_{ij} V_j sin(\Theta_i - \Theta_j) - B_{ij} V_j cos(\Theta_i - \Theta_j))$$

In this case, $P_i$ and $Q_i$ stand for bus “i” active and reactive powers, respectively. The voltage level and phase angle at bus i, respectively, are $V_i$ and $\Theta_i$. The conductance and susceptance between bus i and j, respectively, are $G_{ij}$ and $B_{ij}$.

The equations explain how active and reactive power are distributed among the buses. A bus active power injection equals its active power consumption plus its active power transmission to other buses. A bus reactive power consumption plus its reactive power transmission to other buses equals the reactive power that is pumped into the bus.

These equations can be solved using a variety of numerical approaches, including the Newton Raphson and Gauss Seidel methods. Important information about the network, such as the active and reactive power flowing through each line of the system and the amplitude and phase angle of the voltage at each bus, can be derived by solving these equations.
Panda power has been used to solve this power flow system and simulations are done in order to observe how SDN behaves when there is a different range of reactive power support available. The code for this analysis is shown in Appendix 2: OPF Code

### 4.2.2 Results of power flow Analysis

Three different cases have been studied as shown below.

**1. SDN with No Reactive power support**

To see the impact on our network without any external support of reactive power providing devices, all DER's are removed and power flow analysis is done on the CIGRE network. Following results were obtained.

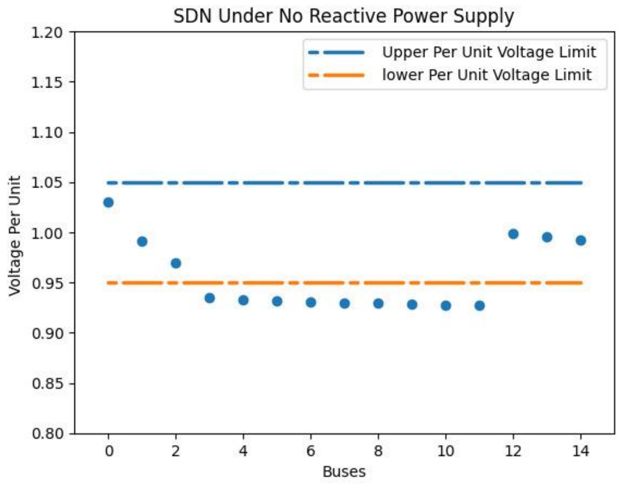


*Figure 9 Voltage Profile with no Reactive Power Support*

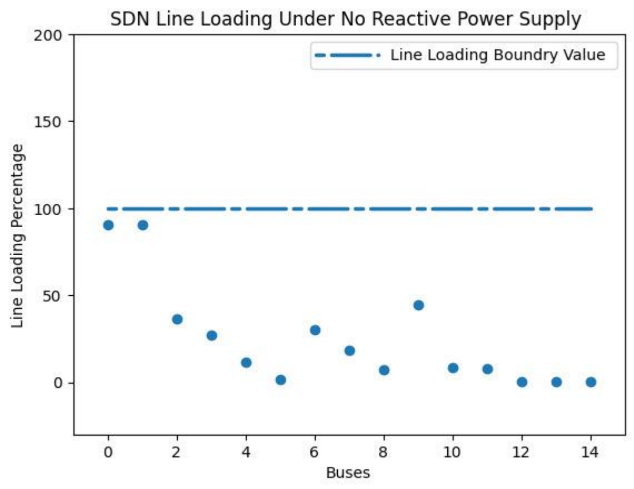


*Figure 10 Loading Percentage with no reactive power support*

**Voltage Instability**

Figure 9 Voltage Profile with no Reactive Power Support shows the voltage profile of the network. The acceptable voltage range is between 0.95 PU to 1.05 PU. In this case there are many buses where under voltage can be observed. Reason is that, in the Network there are heavy industrial loads mounted in which many components are of inductive load. So, with no reactive power support, network stability gets disturbed. Voltage fluctuations and voltage instability arise in the absence of reactive power supply. Due to this instability, the network may fail, resulting in data loss and downtime.

**High loading of Network**

Figure 10 Loading Percentage with no reactive power support shows the loading percentage of buses. Although loading percentages are below 100% but still very high on some buses. This is happening because of no external reactive power support which is causing voltage fluctuations and resulting in high loadings of buses. Without support of reactive power, the network's components function at greater currents, which could eventually overload them and cause failure. Increased power losses, decreased efficiency, and greater operational expenses can result from overloading.

2. **SDN under Normal DER's injection:**

   In this case all DER's are attached to the network and their active and reactive power support is supplied via smart inverters. Each DER is providing **0.2 MVAR** of reactive power support. This is the case with Normal injection of PV's in which reactive power is set to be not very high for the network.

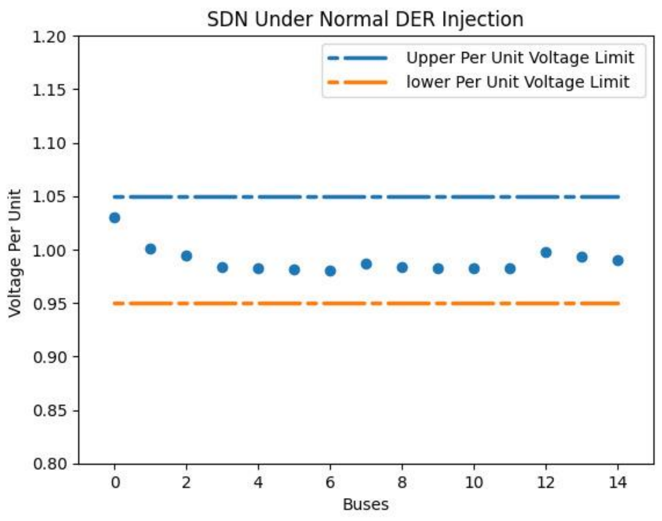


*Figure 11 Voltage Profile with Normal DER Injection*

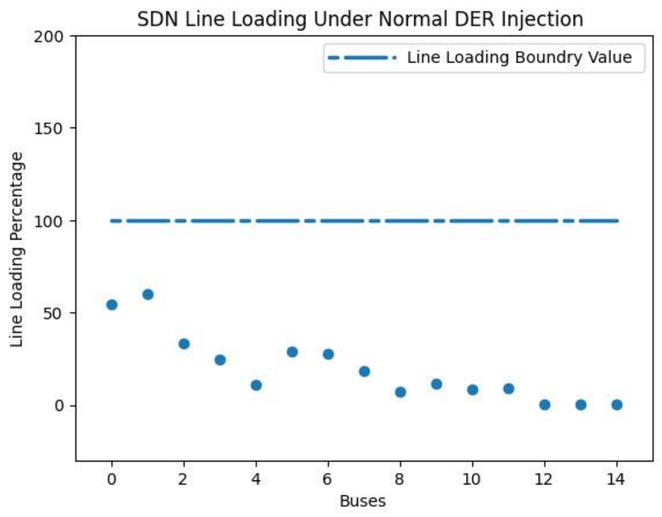


*Figure 12 Loading Percentage with Normal DER Injection*

**Increased Voltage Stability**

Figure 11 Voltage Profile with Normal DER Injection shows that voltage profile of network is within range i.e. 0.95 PU to 1.05 PU. DER reactive power supply aids in maintaining network voltage stability. The DER also assists in controlling the voltage at the point of connection and lowering voltage fluctuations by supplying reactive power which is needed, keeping the power factor within range. Also, the injection of DER's in the network can enhance performance, and the chance of downtime and data loss can be decreased.

**Increased Network Efficiency**

Figure 12 Loading Percentage with Normal DER Injection shows loading percentage of each bus. Here it can be observed that the loading percentage as compared to case 1 is very less on each bus. Reactive power assistance from DER aids in reducing power losses and enhancing network performance. The DER also reduces the amount of reactive power that must be transferred over long distances by providing reactive power close to the load, lowering losses and improving efficiency.

## 3. SDN under High DER's injection:

As the concept of solar power is very popular these days, a lot of rooftop PV's are being mounted every day. This case covers this aspect of the impact that will be produced on our SDN without an external control mechanism for smart inverters.

Each PV is set to be providing **0.5MVAR** of reactive power support mimicking high injection of PV's in SDN.  After running power flow analysis following results were observed.

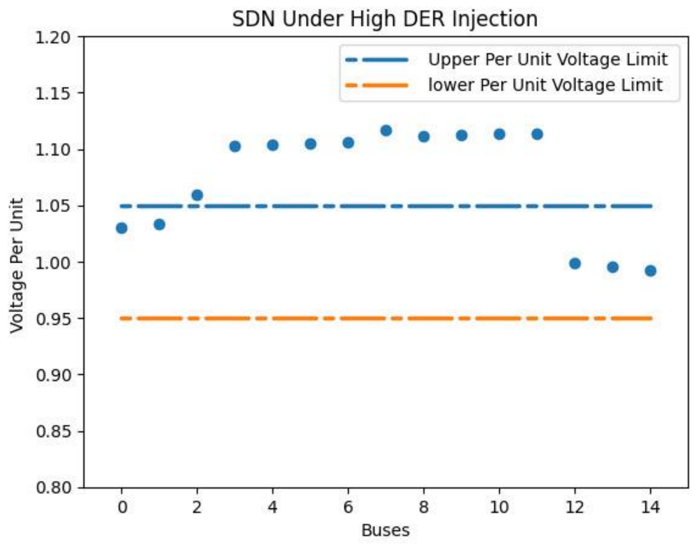


*Figure 13 Voltage Profile with High DER Injection*

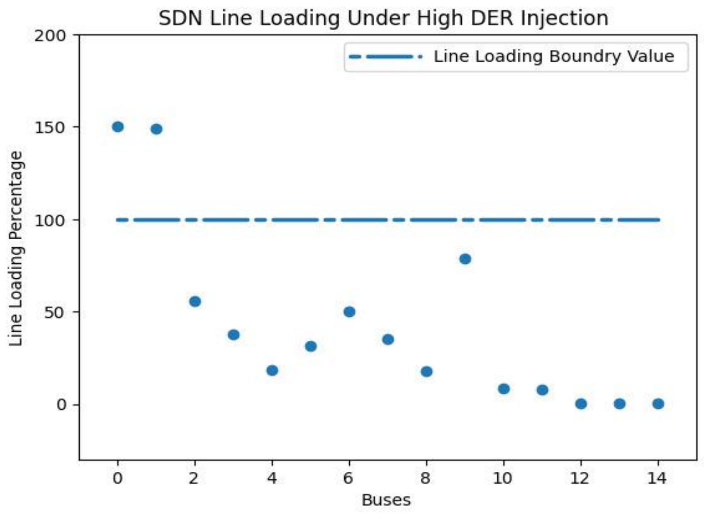


*Figure 14 Loading Percentage under high DER Injection*

**Over Voltage**

By observing Figure 13 Voltage Profile with High DER Injection showing voltage profile of buses under high PV injection following results can be deduced. Overvoltage and voltage instability can result from injecting the network with extremely high quantities of reactive power. It not only disturbs the power factor but also causes capacitor switching transients which leads to voltage spikes as well. Overvoltage can harm equipment and lower the level of service that users receive.

**Extremely High Loading**

Figure 14 Loading Percentage under high DER Injection shows that under this case there is huge overloading on most of the buses. As there are a lot of voltage fluctuations under high injection, the loading of the bus is also not stable. High loading results in very high losses, burning and damaging of equipment.

**Conclusion**

From all the three cases discussed above, it can be clearly observed that reactive power support is very important for the Distribution network in order to have stable voltage profile throughout the network but at the same point if there is too much injection of reactive power then rather benefiting the system it results in over voltages causing damage to the network. So, there should be proper control methodology defined to have bounded reactive power support from DER's as per requirement.

## 4.3. OPF implementation for Data Collection

To train a machine learning model it is important to have a good amount of controlled dataset. We will use optimal power flow analysis to acquire the amount of active and reactive powers which needs to be injected into the system in order to have smooth voltage profile and least loadings on the bus. Also, at this point total 13 PV's (Appendix 1: DER's rating [16, 30]) are attached to show high PV injection in the network which will be balanced by OPF. Appendix 2: OPF Code shows the code for OPF developed using panda power as main tool.

### 4.3.1 OPF Details

The optimum reactive power injection from DERs, such as solar photovoltaic (PV) systems, wind turbines, and energy storage systems, can be determined using OPF. Voltage stability and voltage failure in the power system are both prevented by reactive power injection.

To calculate reactive power injection from DER using OPF following steps have been taken.

- **System Model Defining**: In our case CIGRE Network has been defined in the previous section. All network constraints such as voltage limits and transmission line capacity limits are also included.

- **Adding DER's into System**: In this step DER's and external generators are added into the system. In our case we have 13 DER's as external sources of reactive power. Their capacities and reactive power capability is attached in Appendix 1: DER's rating [16, 30].

- **Formulation of objective Function**: This is one of the critical steps in OPF in which overall system cost is reduced while satisfying the operational constraints.

### 4.3.2 Mathematical Modelling for OPF

In this thesis the objective function $F(\underline{p}, \underline{u})$ is formulated by using External penalty function [16]. Constrained issues are converted to unconstrained problems using the arbitrary penalty function. The objective function is converted to an unconstrained objective function by using the EPF to integrate bus voltage constraints [30].

**Equation A:**

$$min_{\underline{p}}\ F(\underline{p}, \underline{u}) = \ f(\underline{p}) + g(\underline{p}\ , \underline{u})$$

Equation A shows the objective function for our OPF problem which is linear combination of $f(\underline{p})$ and the penalty function $g(\underline{p}\ , \underline{u})$.

The primary goal of solving this OPF problem is to determine the DER's reactive power set points. Variable "p" in the equation B shows the vector of reactive power obtained from NDER's.

**Equation B:**

$$\underline{p} = [Q_{DER1} + Q_{DER2} \ldots\ldots\ldots\ldots\ldots Q_{DERN}]^{\ T}$$

For the reactive power output from DER, there is a limit defined to keep it between certain limits. Limits are applied to apparent power ($S_{DERk}$) and active power ($P_{DERk}$) of DER's. Equation C shows these bounds.

**Equation C:**

$$Q_{DERk} \leq \pm\sqrt{(S_{DERk})^2 - (P_{DERK})^2} \quad \forall\, k = 1,\ldots\ldots, N_{DER}$$

The second part of objective function is $g(\underline{p}\,, \underline{u})$ which shows the penalty function. There will be a penalty charged to the system if the voltages go beyond $V_{min}$ *and* $V_{max}$ range. There are many rules for setting this range in the Distribution Network. In our case we can define these limits in the code and if there is violation then the penalty variable will carry a value.

**Equation D:**

$$g(\underline{p}\,, \underline{u}) = K_p\left[\sum_{s=1}^{N_{bus}} M_s(\underline{p})\right]$$

Where Kp is the penalty multiplier. It makes the value of objective function go very high in case there is voltage violation during the optimization process. In panda power this can be observed by non-convergence of OPF. The penalization functions are further defined by Equation E. [30]

**Equation E:**

$$M_s(\underline{p}) = \left\{0, V_s{}^{min} \leq |V_s| \leq V_s{}^{max} \quad \& \quad 1, otherwise\right\} \quad \forall\, s = 1,2,\ldots\ldots N_{buses}$$

In summary Equation A shows our objective function and Equation C shows the constraints for optimization.

### 4.3.3 Results and discussion

Following Results were obtained after running OPF on our network.

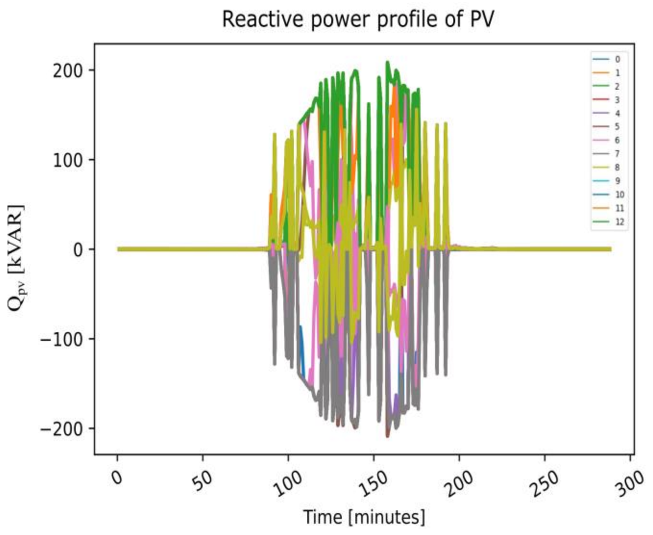


*Figure 15 Reactive Profile of all PV's*

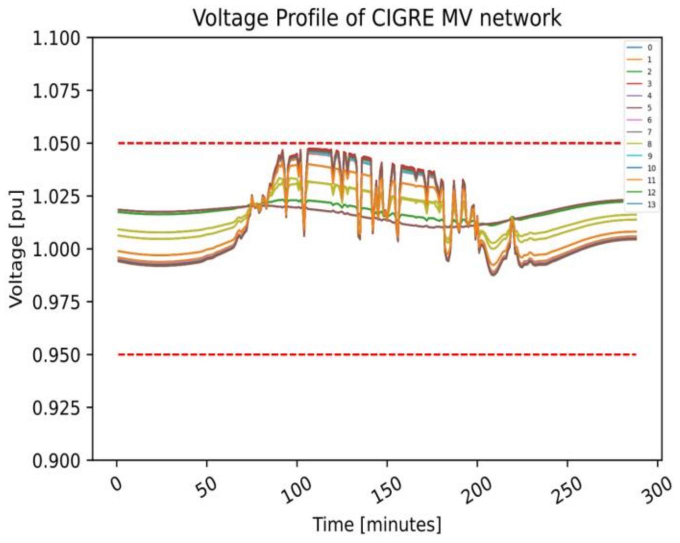


*Figure 16 Voltage Profile of CIGRE Network for all Buses*

It can be observed from Figure 16 Voltage Profile of CIGRE Network for all Buses that the voltage profile of all the buses is lying within constraints because the OPF problem [30] which we formulated earlier has optimized the injection of reactive power in the grid in order to keep a stable voltage throughout the network. From Figure 15 Reactive Profile of all PV's we can see that there is a lot of variation between reactive power profiles of each PV ranging from +200 KVAR to -200KVAR .The reason is at some points PV’s are injecting reactive power into the Network and at some points reactive power is being absorbed to keep the voltage profile of the whole network between the ranges of 0.95 PU to 1.05 PU.

The initial data used for this simulation is obtained by sensitivity analysis as discussed in the research paper [16]. Furthermore, changes in the load profile helped to generate more data for training our ML algorithms which will be discussed in upcoming sections.

## 4.4 ML Algorithm Implementations

Reactive power plays a critical role in power system operations, as it is required to maintain voltage levels and ensure stable and efficient energy transfer. Accurate prediction of reactive power is essential for the optimal control and management of power systems. Traditional methods for predicting reactive power often rely on mathematical models based on power system parameters, which can be complex and computationally intensive. In recent years, machine learning techniques have shown promising results for time series prediction, offering a more efficient and accurate alternative to traditional methods. In this thesis, we explore the use of three machine learning models - Convolutional Neural Network (CNN), Facebook Prophet, and Long Short-Term Memory (LSTM) - for reactive power prediction and compare their performance to traditional models.

**Overview**

The main objective of this thesis is to evaluate the performance of different machine learning models for reactive power prediction. In the first section, we provide an overview of the process we followed to implement these models. This includes data acquisition, preprocessing, feature selection, model development, and evaluation. We then present three machine learning models - CNN, Neural Prophet, and LSTM - and describe their mathematical modeling and how they are typically used in machine learning. For each model, we explain how we adapted it to predict reactive power and any modifications made to the model architecture or parameters. We also provide details about the input data used in each model and how it was preprocessed. In the final section, we present the results of our analysis and discuss the strengths and weaknesses of each model. We also compare the performance of the machine learning models to traditional models and highlight any interesting or unexpected findings. Overall, this thesis contributes to the development of more efficient and accurate methods for reactive power prediction, which can aid in the optimal control and management of power systems.

**Data acquisition**

The initial data set was obtained through sensitivity analysis, a method that involves varying input variables within a defined range and observing the resulting changes in a mathematical model's output. Once the initial data set was acquired by sensitivity analysis [16], optimal power flow analysis was performed to obtain additional data. Optimal power flow analysis is a method used in power systems engineering to optimize the operation of a power system and involves solving an optimization problem to determine the optimal values for the system's control variables.
Together, these two methods were used to acquire the data needed for this project.

**Preprocessing**

The preprocessing steps in this process involve converting Load P and Q values into MVAR and concatenating all inputs into a single array, which includes Loads Active Power, Loads Reactive Power, and PVs reactive power. These steps are necessary to ensure that the input data is in the correct format for use in the machine learning model.

**Feature Selection**

The output values in this process are scaled by subtracting their means and dividing by a specific standard deviation. This step is important for feature selection, as it helps to ensure that the machine learning model can converge easily. By scaling the output values, the model can more effectively identify which features are most relevant to predicting the output.

In summary, the preprocessing steps involve converting Load P and Q values and concatenating inputs, while the feature selection step involves scaling output values to facilitate the machine learning model's convergence and effective feature selection.

### 4.4.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have gained widespread popularity in recent years, especially in image recognition tasks. However, they have also demonstrated their effectiveness in processing sequential data such as time series. In this work, we explore the use of CNNs to predict reactive power in power systems.

The working operation behind a CNN is to extract features from the input data using convolutional layers. In the case of time series, the input is a sequence of data points over time. The convolutional layers apply filters to the sequence, which perform a local convolution operation, capturing local patterns from the sequence. By applying multiple filters with different weights, the convolutional layer can extract a variety of features from the input sequence.
The output from the convolutional layers is then passed through one or more fully connected layers, which perform classification or regression, based on the extracted features. In our case, the output is a predicted value of the reactive power.

The entire network is trained using backpropagation to minimize the loss function, which measures the difference between the predicted output and the true output. By iteratively adjusting the weights of the network, the model learns to accurately predict the reactive power [42].
Our motivation for using CNNs in this work is the fact that real-world time series often exhibit local patterns and dependencies that can be captured by convolutional layers. This has led to the development of a variety of CNN architectures for time series, including 1D and 2D CNNs, as well as more complex architectures such as Wave Net and TCN.

The CNN architecture which we used in our use case consists of 3 hidden layers with a total 7 layers in it. Where the activation functions used are sigmoid. The reason to choose the sigmoid in this scenario is because sometimes it gives very consistent results as per our use case. The Neurons we selected are following:

- 32 in first layer of CNN.
- 26 in second layer of CNN.
- 13 in the last hidden layer.

Table 3 Implemented Multi-Layer Perceptron (MLP) Architecture

| Layer | Input | In Size | K | S | Activation | Nodes | Output Size |
|---|---|---|---|---|---|---|---|
| Dense-1 | Input Array | 39x1 | - | - | sigmoid | 39 | 32x1 |
| Dropout-1 (50%) | Dense-1 | 32x1 | - | - | - | - | 32x1 |
| Dense-2 | Dropout-1 | 32x1 | - | - | sigmoid | 26 | 26x1 |
| Dropout-2 (50%) | Dense-2 | 26x1 | - | - | - | - | 26x1 |
| Dense-3 | Dropout-2 | 26x1 | - | - | sigmoid | 18 | 18x1 |
| Dropout-3 (50%) | Dense-3 | 18x1 | - | - | - | - | 18x1 |
| Dense-4 | Dropout-3 | 18x1 | - | - | - | 13 | 13x1 |

Input Array size is 39 because we have concatenated data of 13 PV's Reactive power at each time step with load active and reactive powers values. As loads are attached to only 13 buses just like PV's, so total array size becomes 39 which is our input size.

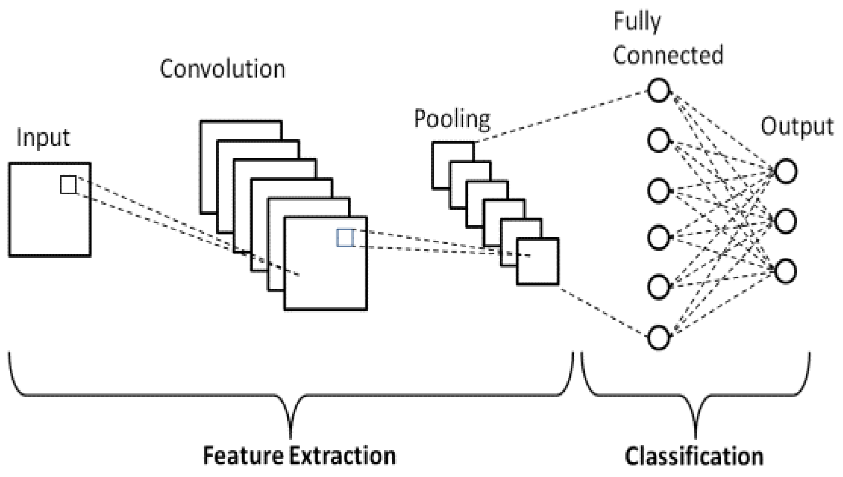


Figure 17 CNN Basic Architecture [50]

Figure 17 CNN Basic Architecture [50] shows the architecture of the CNN, which clearly represents the working of CNN architecture. In our use case, the input consists of time series data and then it goes for feature engineering and then passed to the fully connected layer for performing the operation of forecasting the future results [50].

**Mathematical Modeling:**

- **Input data representation**

  In time series analysis, the input data is usually a sequence of data points over time. These data points can be represented as a one-dimensional array or matrix, depending on the type of CNN architecture used.

- **Convolution operation**

  The convolution operation involves sliding a filter over the input data and computing the dot product between the filter and the input at each position. Mathematically, this can be expressed as [42]:

  $$Output_i = Sum_j \left(Input_{(i+j)} \times Filter_j\right)$$

  Where Output is the output of the convolution operation, Input is the input data, Filter is the filter being applied, and i and j are the indices of the output and filter, respectively.

- **Activation function**

  After the convolution operation, an activation function is applied to the output to introduce nonlinearity into the model. Common activation functions used in CNNs for time series include ReLU (Rectified Linear Unit), Tanh (Hyperbolic Tangent), and Sigmoid [52].

- **Pooling operation**

  The pooling operation involves reducing the dimensionality of the output by aggregating nearby values. Common pooling operations used in CNNs for time series include Max pooling and Average pooling.

- **Fully connected layer**

  The collected features are then used for classification or regression in one or more fully connected layers that follow the convolutional and pooling layers [42]. Fully-connected layers have an output that is calculated as:

$$Output = activation\_function(Wx + b)$$

  Where W is the weight matrix, x is the input vector, b is the bias vector, and the activation function is usually a nonlinear function such as Re LU or Sigmoid.

- **Loss function**

  The loss function measures the difference between the predicted output and the true output. Common loss functions used in CNNs for time series include Mean Squared Error (MSE) and Cross-Entropy loss.

- **Backpropagation**

  Backpropagation is used to train the entire network by calculating the loss function's gradients with regard to the network's weights and biases. Stochastic Gradient Descent (SGD) and Adam are two examples of optimization algorithms that can be used to update the weights and biases based on these gradients.

**Evaluation on the use of CNN in our work:**

In our use case the problem of reactive power forecasting in power systems is a critical task that helps in maintaining the stability and reliability of the grid. CNNs have been used for this task due to their ability to capture local patterns and dependencies in time series data. The input to the CNN is usually a sequence of historical data, and the output is the predicted reactive power at a future time step. The CNN consists of several layers, including convolutional layers, activation layers, pooling layers, and fully connected layers.

The total evaluation of our CNN is bit better for several reasons due to the flexible functionality of the handling the time series data but when it comes towards the Power Sector it has many aspects of continuity for the sake of argument, we say the continuous voltage which is a best analogy for CNN to solve in order to get the better results. The other reason is to work efficiently is that it is little better than the conventional methods of the time series data like the other choices like ARIMA and SARIMA in relevant use cases of Reactive Power.

### 4.4.2 Deep Neural Prophet

In this work, we have used the Deep Neural Prophet model to predict reactive power. This model is an advanced version of the Neural Prophet time-series forecasting model that incorporates deep learning techniques to improve the forecasting accuracy of time-series data. The Deep Neural Prophet model builds upon the architecture of the original Neural Prophet model and extends it to include multiple hidden layers with many neurons.

The model architecture is composed of three main components: an input layer, a sequence-to-sequence RNN layer, and a feedforward neural network (FNN) output layer.

The input layer receives the time-series data of reactive power and converts it into a format that can be processed by the RNN layer. The RNN layer captures the temporal dependencies in the time-series data and generates a sequence of hidden states, which are then passed to the FNN output layer. The FNN output layer predicts the future values of reactive power based on the hidden states generated by the RNN layer.

One of the strengths of the Deep Neural Prophet model is its ability to handle multiple seasonality and trend change points in reactive power data. The model achieves this by using a combination of Fourier series and linear regression to model seasonality, and by incorporating a piecewise linear function to capture trend change points [52]. This makes the Deep Neural Prophet model well-suited for predicting reactive power in dynamic systems with complex and changing patterns.

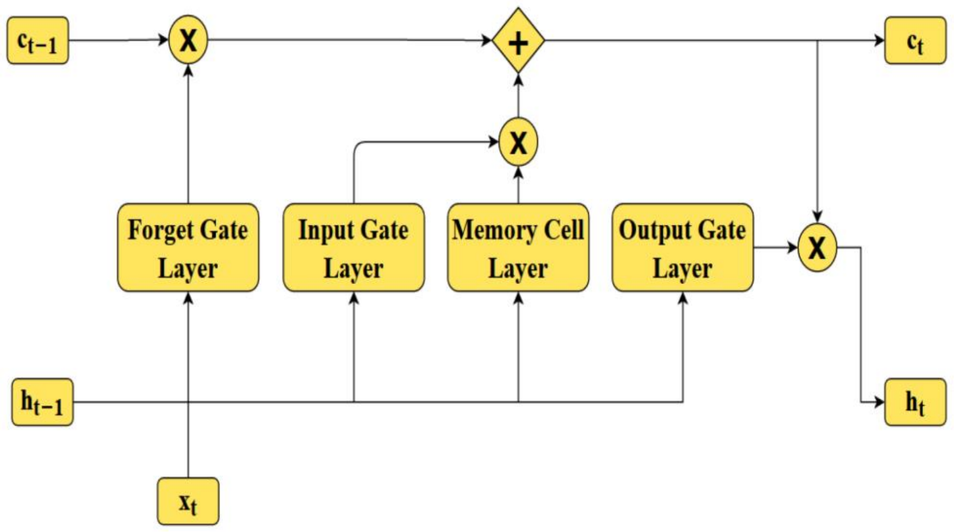


*Figure 18 Basic Structure on DNP model [48]*

Figure 18 Basic Structure on DNP model shows the basic working of Deep Neural Prophet model that we have used in our work for predicting the reactive power.

The model is defined as follows:

$$Y(t) \;=\; f(Y(t-1), Y(t-2), \ldots, Y(t-n), X(t), X(t-1), \ldots, X(t-m))$$

where *Y(t)* is the target variable at time t, *Y(t-1), Y(t-2), ..., Y(t-n)* are lagged values of the target variable, *X(t)* is the input variable at time t, and *X(t-1), ..., X(t-m)* are lagged values of the input variable. The function f represents the neural network architecture that maps [48].

Reactive power forecasting is a critical component in maintaining the stability and efficiency of Smart Distribution Networks (SDN). In recent years, machine learning techniques have emerged as a powerful tool for forecasting reactive power in SDN networks.

**Evaluation:**

The main reason for using the neural prophet is its splendid property of handling the multivariate use cases especially like the Reactive Power Forecasting. The Performance is mostly better than the

CNN results due to several aspects but the main issue with the Neural Prophet is that it works accurately when we give it more and large data batches so in some cases it gives lower performance than the other ML Methodologies.

The Neural Prophet model is a deep learning-based model that can capture complex temporal patterns in time-series data. Using this model, we were able to forecast the reactive power in SDN networks by considering the various parameters that affect it, such as voltage, current, and power factor. The results of this study demonstrate that the Neural Prophet model is an effective and efficient tool for reactive power forecasting in SDN networks and can contribute significantly to the overall stability and efficiency of the network.

### 4.4.3 Long Short-Term Memory (LSTM) Model

Time-series forecasting is a common use of LSTM (Long Short-Term Memory), a form of recurrent neural network (RNN). To overcome the difficulty that conventional RNNs have in learning long-term dependencies in time-series data, LSTM was developed. The input gate, forget gate, and output gate that make up LSTM's architecture provide the model the ability to learn and forget information selectively over time.

By modeling the data's temporal dependencies, LSTM may produce reliable forecasts of future values for time-series analysis. The model accepts a series of historical data as input and produces a series of forecasts. For time-series forecasting, where the length of the time-series may vary, LSTM's capacity to handle variable-length sequences is crucial.

Time-series forecasting tasks such as predicting stock prices, energy demand, and the weather forecasting, have all benefited from the use of LSTM. Time-series forecasting methods like ARIMA and exponential smoothing have been demonstrated to be inferior to this method. On the other hand, LSTM is computationally demanding, especially for large datasets, and requires careful tuning of hyper parameters. Even yet, LSTM continues to be a well-liked and efficient option for many time-series forecasting applications, especially in light of the expanding availability of computational resources and the ever-increasing demand for precise time-series forecasting.

.

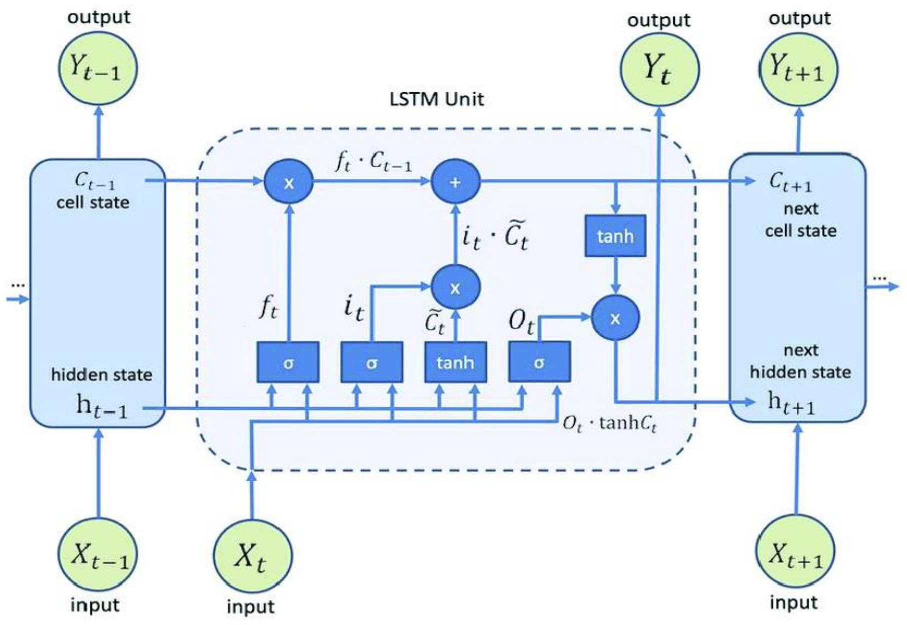


*Figure 19 Mathematical Model of LSTM [53]*

Mathematically as shown in Figure 19 Mathematical Model of LSTM [53], an LSTM network consists of a sequence of cells, each with a hidden state $h_t$ and a cell state $C_t$ [47]. The cell state is updated using a combination of three gates: the input gate $i_t$, the forget gate $f_t$, and the output gate $O_t$. These gates are responsible for controlling the flow of information into and out of the cell.

The input gate $i_t$ determines how much of the new input $x_t$ should be added to the cell state. It is computed as:

$$i_t = \sigma(W_i\,[h_{\{t-1\}}, x_t] + b_i)$$

Where $W_i$ and $b_i$ are learnable weights and biases, and σ is the sigmoid activation function.

The forget gate $f_t$ determines how much of the previous cell state $C_{t-1}$ should be retained. It is computed as:

$$f_t = \sigma(W_f\,[[h_{\{t-1\}}, x_t] + b_f)$$

The output gate $o_t$ determines how much of the cell state should be used to compute the hidden state $h_t$. It is computed as:

$$O_t = \sigma(W_o [h_{\{t-1\}}, x_t] + b_o)$$

The new cell state $c_t$ is computed as a combination of the input and forget gates and the previous cell state [47]:

$$C_t = f_t \times C_{\{t-1\}} + i_t \times tanh\left(W_c \left[h_{\{t-1\}}, x_t\right] + b_c\right)$$

Where $W_c$ and $b_c$ are learnable weights and biases, and tanh is the hyperbolic tangent activation function.

Finally, the new hidden state $h_t$ is computed as a combination of the output gate and the updated cell state:

$$h_t = h_t \times tanh(c_t)$$

LSTM's gating mechanism allows it to selectively retain and forget information over time [53], making it well-suited for time-series forecasting tasks where long-term dependencies are important. I proposed a machine learning approach based on the LSTM (Long Short-Term Memory) neural network model for reactive power forecasting in SDN networks. The proposed model considers the various parameters that affect reactive power, such as voltage, current, and power factor, and captures the temporal dependencies in the time-series data.

**Evaluation**

The LSTM model was trained on historical data of reactive power consumption and the corresponding parameters. The trained model was then used to forecast the reactive power consumption for future time periods. The results of the study showed that the LSTM model was able to accurately forecast reactive power consumption in SDN networks with a high degree of accuracy.

The LSTM in our use case consists of the following architecture. It consists of two layers because the data was not too much which can complete the need of LSTM true potential so in such cases, we must merge very few numbers of Layers in it. The Neurons are 50 which are fine for our use case so it can work properly for our forecasting results with the 50 epochs in its training because this number of epochs are sufficient in such cases.

Compared to traditional statistical methods such as ARIMA and exponential smoothing, the proposed LSTM-based approach showed superior performance, especially in cases where the reactive power consumption is highly variable over time. The results of this study demonstrate the potential of machine learning models such as LSTM for reactive power forecasting.

# 5. Results and Discussions

This chapter presents all the results obtained after implementation of ML algorithms for prediction of reactive power. There is a section highlighting key points deduced from results and in last there is a comparative analysis of ML versus conventional approach used for forecasting which covers final objective of thesis.

## 5.1 CNN Results

Results obtained after implementing Computational neural network algorithm are shown below. Figure 20 and 21 represent predicted vs actual reactive powers for 13 PV's which are available in our network. Figure 22 CNN Training and validation loss represents total model loss of CNN showing efficiency for this data set. Code for this algorithm is shown in Appendix 3: CNN Code

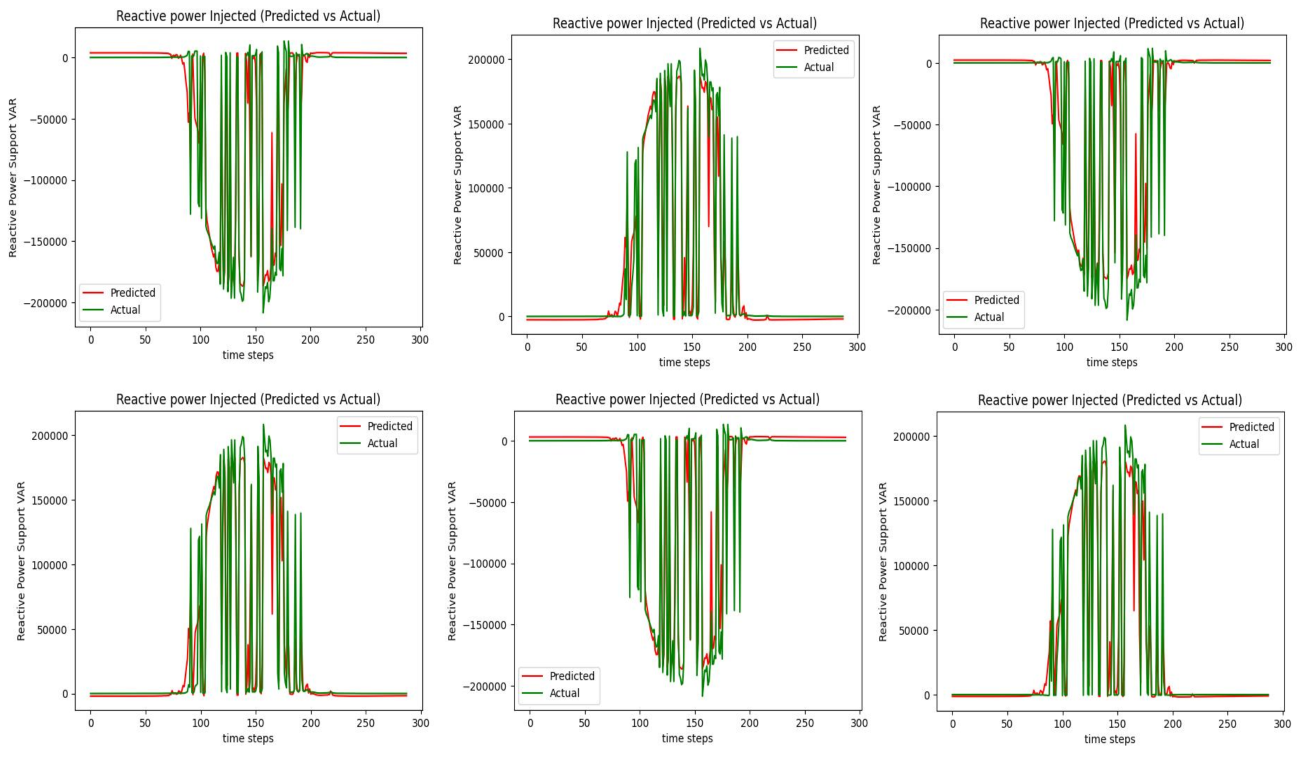


*Figure 20 Predicted vs Actual Reactive Powers for PV 1 to 6 by CNN*

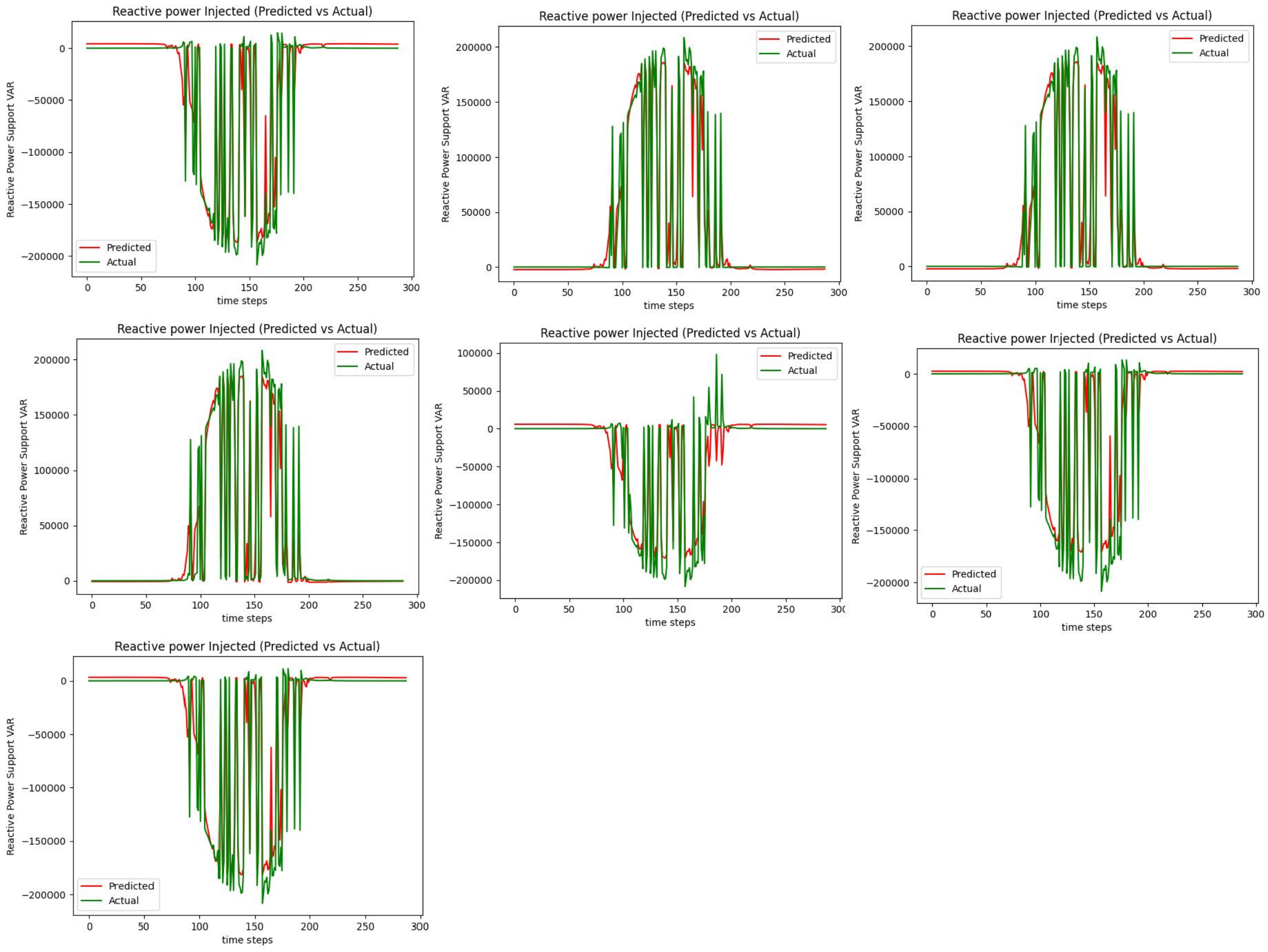


*Figure 21 Predicted vs Actual Reactive Powers for PV 7 to 13 by CNN*

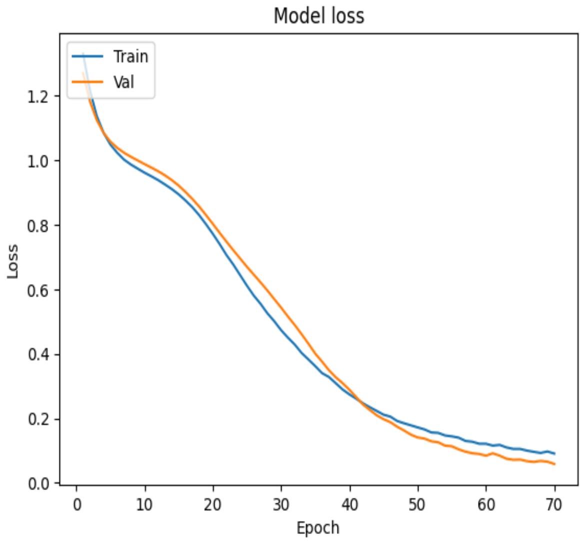
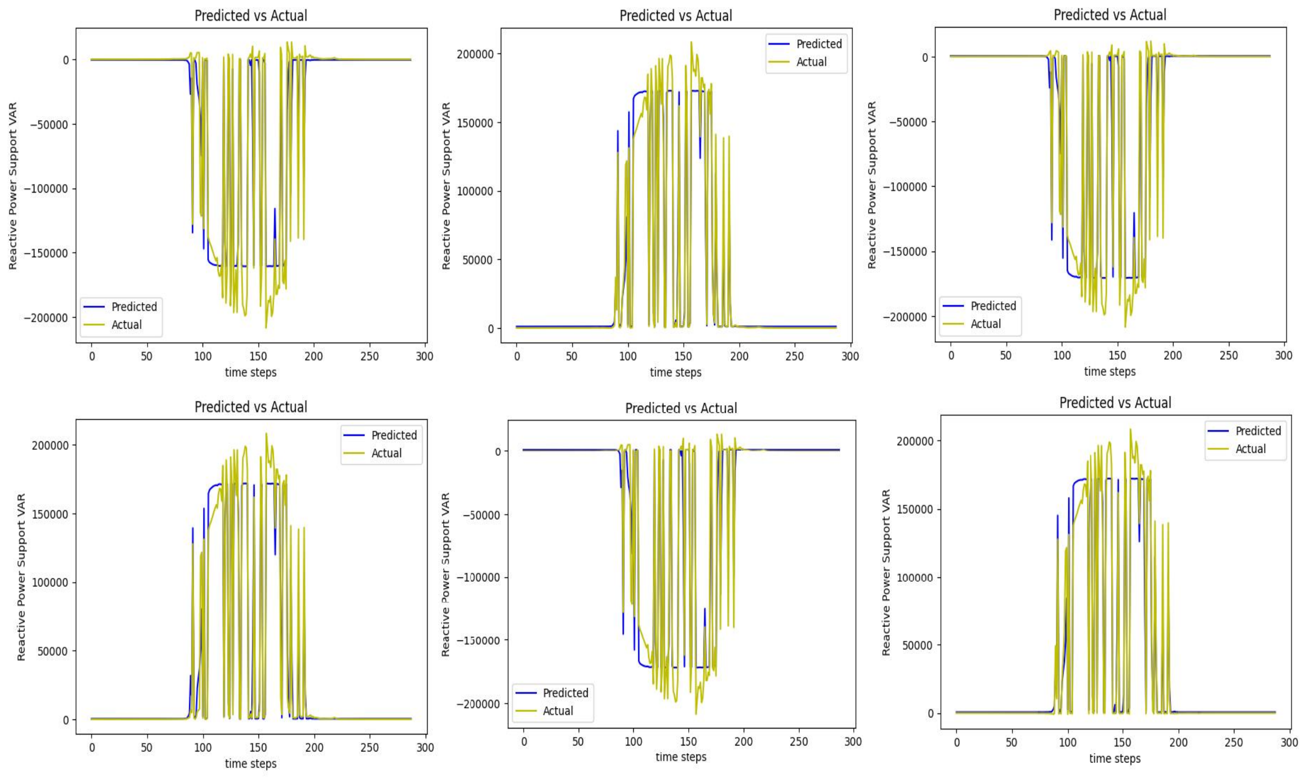


*Figure 22 CNN Training and validation loss*

## 5.2 LSTM Results

The results of applying the LSTM algorithm to our dataset are displayed below. Predicted vs actual reactive powers for 13 PVs that are accessible in our network are shown in Figure 23 Predicted and Actual Reactive Powers for PV 1 to 13 by LSTM. Figure 24: LSTM training and validation loss shows the LSTM's overall model loss and efficiency for the given data set. Code for this algorithm is shown in Appendix 5: LSTM Code



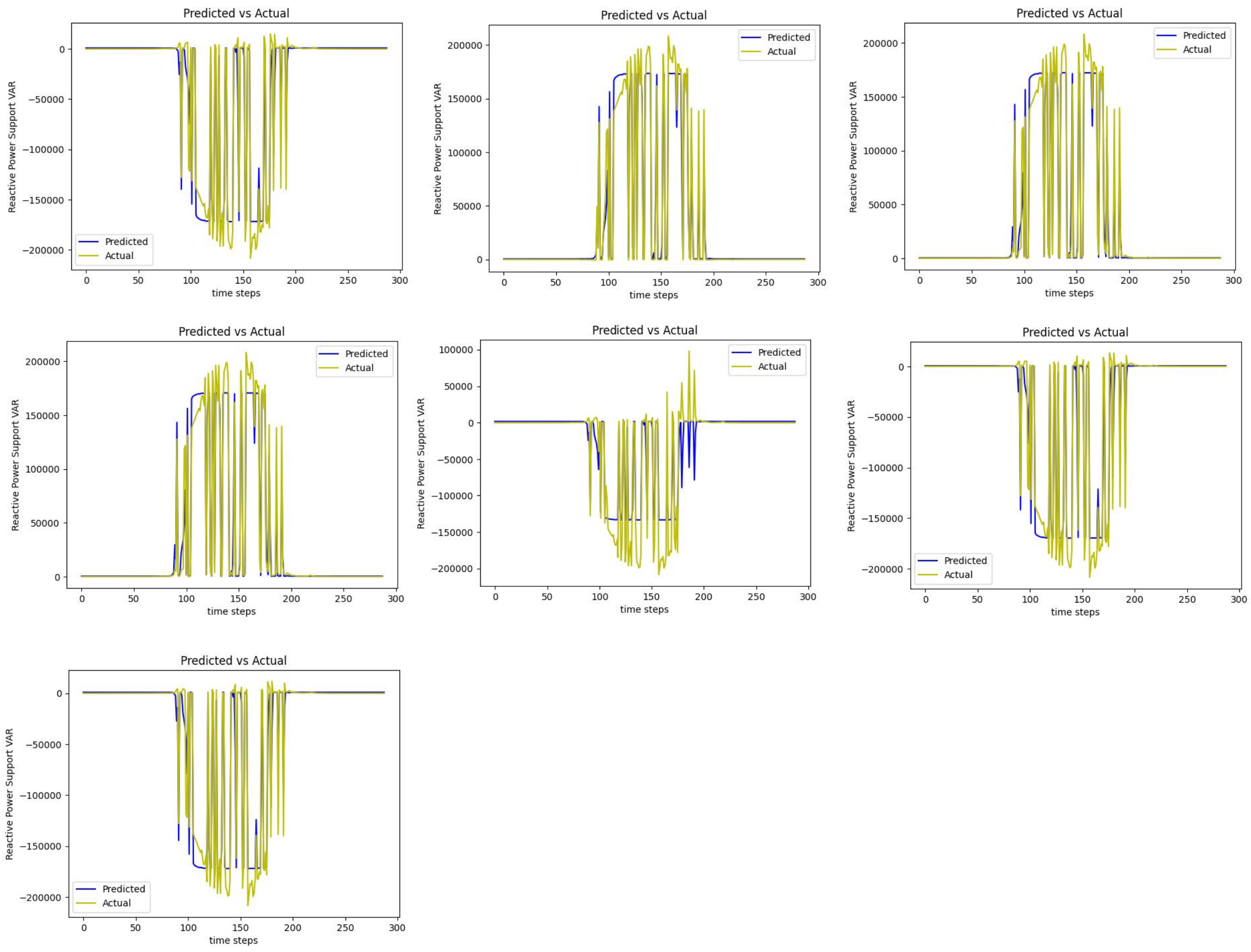


*Figure 23 Predicted and Actual Reactive Powers for PV 1 to 13 by LSTM*

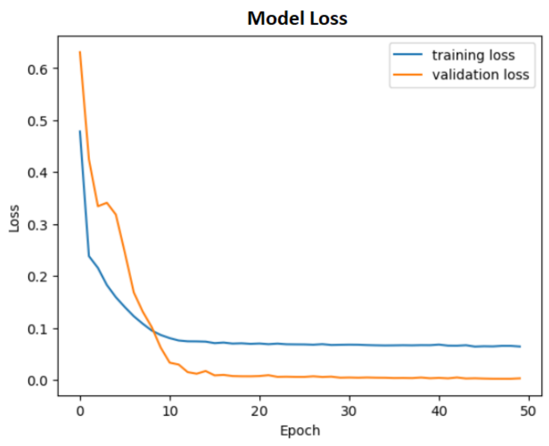
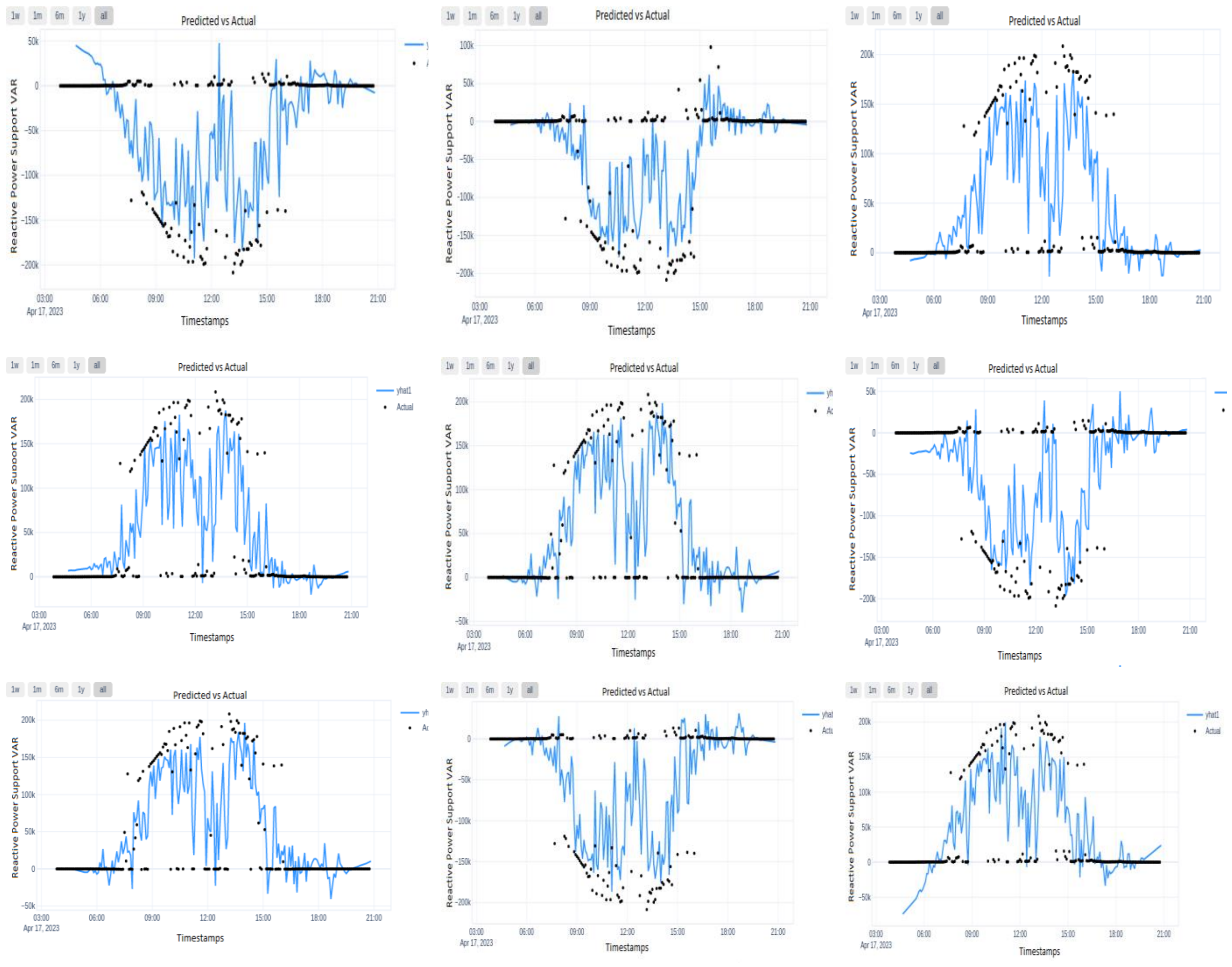


*Figure 24: LSTM training and validation loss*

## 5.3 Neural Prophet

Finally Neural prophet Algorithm is implemented and the results from this algorithm are shown below. Predicted vs. actual reactive powers for 13 PVs that are accessible in our network are shown in Figure 25 Predicted and Actual Reactive Powers for PVs by Neural Prophet. Figure 26 shows training loss of this algorithm. Code for this algorithm is shown in Appendix 4: Neural Prophet Code



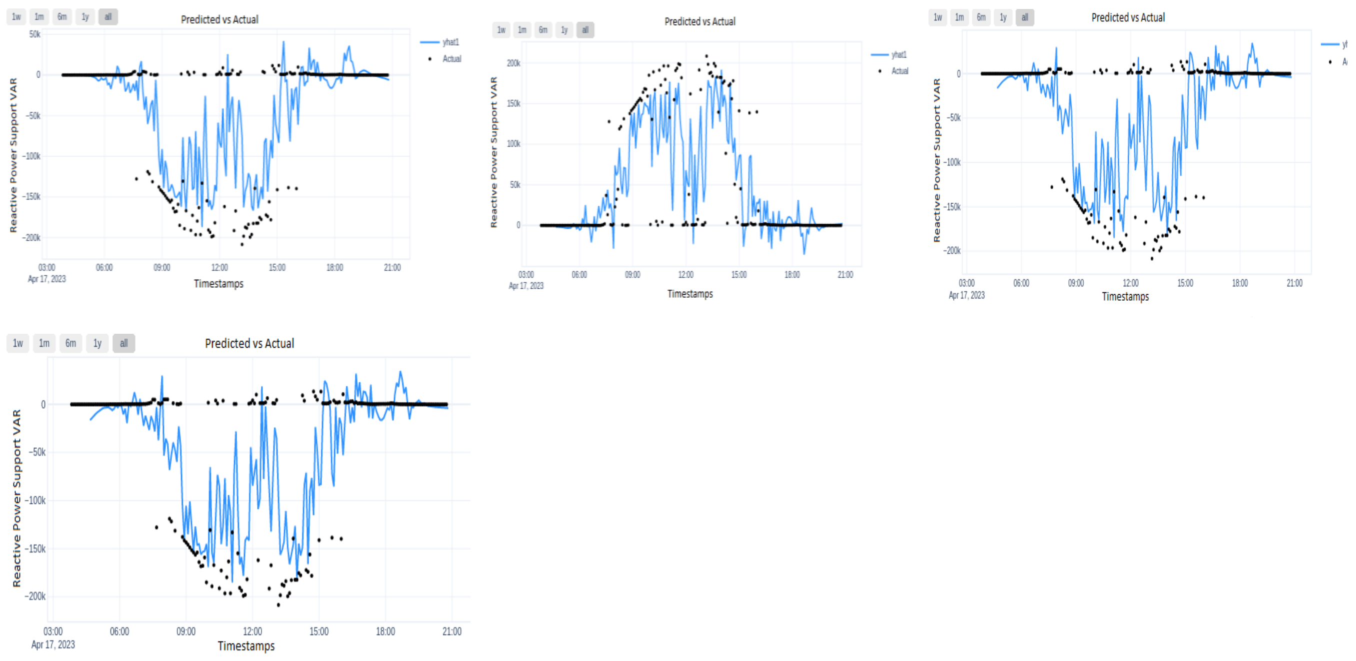


*Figure 25 Predicted and Actual Reactive Powers for PVs by Neural Prophet*

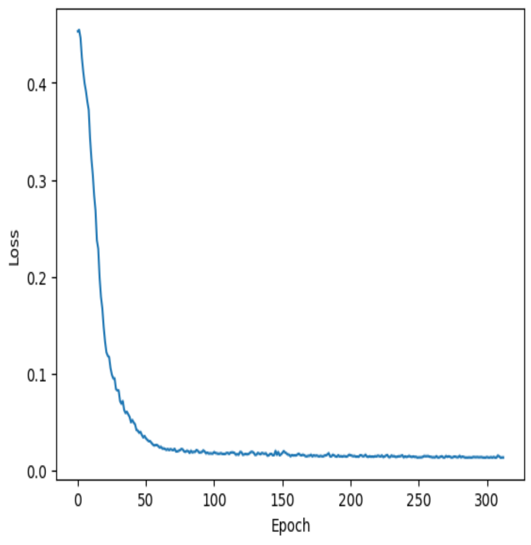


*Figure 26 Training and Loss of Neural Prophet*

## 5.4 Performance Comparison of Algorithms

Table 4: Overview of Algorithms Performance shows the combined performance of all Algorithms. There are some important terms to note here i.e. MAE showing mean absolute error, MSE showing mean square root error and RMSE depicting root mean square root error. It can be observed that LSTM outperforms all other algorithms. More details are given in next section

*Table 4: Overview of Algorithms Performance*

| **Model** | **MAE** | **MSE** | **RMSE** |
|---|---|---|---|
| **CNN** | 0.31 | 0.00033 | ~0.0078 |
| **LSTM** | 0.011 | 0.00002 | ~0.0004 |
| **Neural Prophet** | 1.06 | 1.139 | 1.067 |

## 5.5 Result Discussions

Figure 20 till 25 (excluding figure 22 and 24) shows results for all three algorithms depicting Predicted and actual reactive powers for all 13 PV's. The actual reactive power profile is basically the data which was generated from optimal power flow analysis and on top of that predictive profile is shown with different color.

CNN and LSTM results show similarities in their pattern because despite of having some fundamental differences both algorithms are good for processing sequential data. In the thesis we have used 288-time steps data with 5 minutes interval in each step mimicking one day. Data for a month has been generated for this study. If we observe figures, there are some spikes in predicted values for each PV's reactive power and there is no perfect overfitting. The main reason to that is we have very small data set and to observe best training results years of data is required. It was not practical to generate this big chunk of data for this thesis so simulations resulting on small data shows variations.

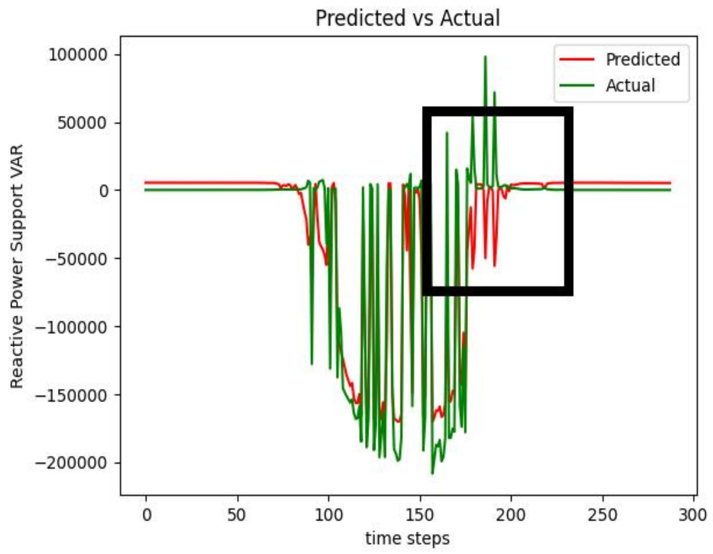


*Figure 27 Actual vs Predicted Reactive Power of 3rd PV*

Another careful result which can be deduced by observing negative spike from the 3rd PV predicted value in Figure 27 Actual vs Predicted Reactive Power of 3rd PV is that it is not only important to have big size of data availability but also Normalization of input data is very important for CNN and LSTM. As we see that there is sudden pattern change in Actual reactive power data i.e., first reactive power was being absorbed and suddenly PV started to deliver Reactive power. In such cases of sharp shifts if the input data is not properly normalized to have mean of zero and standard deviation of one then there will be spikes on swift changes in values. In our case LSTM outperforms CNN because LSTM architecture is very compatible with time series data where CNN is mostly suitable for image processing, object detection and some other similar applications.

Performance of CNN and LTSM algorithms can be seen in Figure 22 CNN Training and validation loss and Figure 24: LSTM training and validation loss. Two important terms to interpret these figures are training loss and validation loss.

### 5.5.1 Training Loss:

The model's performance on the training set of data is measured by the training loss. With the aim of minimizing the training loss, the model's parameters are modified during training in response to the training data. At the end of each training period or batch, the training loss can be determined, and the average training loss for the whole training process is often provided.

### 5.5.2 Validation Loss:

On the other side, the validation loss measures how effectively the model generalizes to fresh, untried data. A subset of the data is kept aside during training as a validation set; this set is not utilized for training but is instead used to assess how well the model performs on data that has not yet been seen. In order to identify overfitting, the validation loss is routinely tracked throughout training. It is calculated using the validation set.

In our case LTSM perform betters and show minimum training and validation loss. The reason is the architecture of LTSM suits our type of application which is time series prediction as discussed above.

Now we address the results of neural prophet. Keeping it separate from other two algorithms is since its working architecture is totally different from those two. Figure 25 Predicted and Actual Reactive Powers for PVs by Neural Prophet shows results obtained from neural prophet Algorithm. Now this algorithm does not perform very well although it is specifically designed to work for time series forecasting models. The model works by using a combination of Fourier series and linear regression to model seasonality, and by incorporating a piecewise linear function to capture trend change points It can capture seasonal patterns, trends, and other time-dependent properties because it is specifically made to handle time-series data. The main reason for low performance in our case is limited data availability. Large batches of data can significantly impact on these results and can be tried in future work. Still it can be observed that for Neural prophet there are no sharp spikes in results indicating its power of computation and efficiency.

### 5.5.3 Overall Performance comparison:

Table 4: Overview of Algorithms Performance shows the performance of individual algorithms when tested for predicted values as compared to the Actual data set. In table there are three parameters given to detect the performance. Each one of those have its pro’s and con’s. MAE is the absolute error so it does not account for negative values so even if there is some negative error

coming, it will take its absolute value and also it is less sensitive to higher error values. On contrary MSE (mean square root error) is highly sensitive to large error values and accounts for both positive and negative values. So among all of those parameters RMSE (root mean square root error) is best to judge the performance of any algorithm because it does not only accounts for positive or negative values but is less biased in case of high values [52].
According to results obtained, LSTM is best performing algorithm among all others and CNN shows slightly lower performance than LSTM because it has lot of architecture similarity with LSTM. Although theoretically neural prophet is more suitable to this application due to its internal layering structure. The main reason for this low performance by neural prophet is due to less data availability. LSTM has the capability to show good performance even at low available data by finding patterns in the data giving good predictions. It is highly recommended that in future work Neural prophet is again used with big chunk of training data to see its actual potential which we were not able to do due to time constraints.

## 5.6 Machine Learning Versus Conventional Approach

In recent years, machine learning (ML) algorithms have been applied to forecast reactive power in power systems. In this section, we compare the performance of three ML algorithms - LSTM, CNN, and Neural Prophet - with respect to forecasting reactive power, in comparison to the conventional methods of Optimal power Flow Analysis (OPF), Droop Control and power factor correction techniques etc.

CNN (Convolutional Neural Network) is a type of neural network that is widely used for image and signal processing applications. In recent years, CNNs have been applied to time series forecasting problems as well. In our study, we applied a 1D CNN model to the reactive power data and compared its performance with the OPF method. The CNN model showed comparable performance to the OPF method.

LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) that has been successfully applied to sequence modeling and time series forecasting. LSTM models can capture long-term dependencies in time series data and are widely used for forecasting applications. In our study, we trained an LSTM model on the reactive power data and evaluated its performance on a test dataset. The LSTM model outperformed CNN in terms of accuracy.

Neural Prophet is a recently developed time series forecasting library that is based on the popular Facebook Prophet forecasting algorithm. Neural Prophet extends Prophet by incorporating neural networks for improved forecasting accuracy. In our study, we used the Neural Prophet library to forecast reactive power and compared its performance with the OPF method. Overall performance of Neural Prophet was lower than other algorithms due to small dataset but model outperformed other methods in terms of accuracy in several aspects such as consistency, convenience etc.

Overall, our results indicate that ML algorithms, such as LSTM, CNN, and Neural Prophet, can be effectively used for reactive power forecasting in power systems. In particular, LSTM and CNN models showed similar performance compared to the conventional OPF method, while the Neural Prophet model showed comparable performance. The ability of ML algorithms to capture nonlinearities and complex patterns in time series data makes them a promising alternative to conventional methods for reactive power forecasting.

One advantage of using ML algorithms for reactive power forecasting is their ability to handle non-linear relationships between input features and output variables. In contrast, conventional methods such as OPF rely on linear approximations and require accurate knowledge of the system parameters, which can be challenging to obtain in practice.

Another advantage of ML algorithms is their ability to adapt to changing system conditions and dynamics. This is particularly important in power systems, where the operating conditions can change rapidly due to changes in load demand and renewable generation. In contrast, conventional methods such as OPF is highly grid model dependent and cannot adapt to changes in the system dynamics.

However, ML algorithms require large amounts of data for training and can be computationally expensive. In addition, the accuracy of ML algorithms is highly dependent on the quality and quantity of the data used for training. On the other hand, conventional methods such as OPF have been widely used in power systems for decades and have a well-established theoretical foundation. In addition, they can be computationally efficient and do not require large amounts of data for training.

### 5.6.1 Performance

The use of machine learning algorithms in the control of power systems has shown encouraging outcomes. Some common machine learning algorithms used for controlling power systems are Long Short-Term Memory, Convolutional Neural Networks, and Neural Prophet. These algorithms can optimize power flow, manage demand response programs, and even predict voltage and frequency in power networks. Using objective functions, machine learning algorithms can optimize control actions in response to dynamic changes in the system's environment..

Compared to traditional control methods such as OPF and droop control, machine learning algorithms can handle uncertainties and dynamic system conditions more effectively. Droop control assumes that all distributed generators have the same droop coefficient, which may not be the case in practice. In contrast, machine learning algorithms can learn from diverse data sources and adapt to changing system conditions, resulting in better performance.

### 5.6.2 Interpretability

Machine learning algorithms suffer from a lack of interpretability, which is one of its drawbacks. The decision-making process behind machine learning algorithms is opaque, therefore it is difficult to understand the final product. In contrast, power system engineers get a clear picture of the reasoning behind OPF and droop control's decisions.

Recent studies have focused on building explainable machine learning approaches that provide insight into the decision-making process of machine learning algorithms, as interpretability is crucial in power system control applications. The LSTM and CNN architectures, for example, have been tweaked to incorporate attention techniques that focus on the most crucial aspects for the output [52].

### 5.6.3 Data Requirements

Machine learning algorithms require large amounts of data to train and optimize the model. In the context of power system control, data can come from various sources such as historical power system data, real-time sensor data, and weather data. However, data collection and management can be a challenge in power system control applications, especially in distributed power systems.

In contrast, traditional control methods such as OPF and droop control require less data and can be more easily implemented in a distributed power system. OPF only requires information about the topology of the power system and operational constraints, while droop control only requires information about the output of distributed generators. However, the performance of these traditional control methods may be limited by their inability to adapt to changing system conditions.

### 5.6.4 Accuracy

Machine learning algorithms have shown high accuracy in power system control applications. For example, LSTM and CNN models have been used to forecast reactive power in power systems with high accuracy. Neural Prophet is another machine learning algorithm that has shown promising results in power system control applications.

In comparison, OPF and droop control can have limited accuracy in dynamic and uncertain conditions. OPF is highly dependent on grid models, making it difficult to redesign for larger grid models. Droop control assumes a constant droop coefficient for all distributed generators, which may not hold true in practice. Machine learning algorithms can adapt to changing system conditions and provide accurate predictions, making them a valuable addition to power system control applications.

### 5.6.5 Robustness

Machine learning algorithms can be robust to uncertainties and disturbances in power systems. For example, LSTM and CNN models have been used to predict voltage and frequency in power systems with high accuracy under various disturbances. Neural Prophet can handle missing data and noise in the input data, making it robust to uncertainties.

In comparison, traditional control methods such as OPF and droop control may not be as robust to uncertainties and disturbances. OPF is highly dependent on grid models and may not perform well under uncertain conditions.

### 5.6.6 Computational Complexity

In power system control applications, where judgments must be made fast, the computational complexity of machine learning methods might be a barrier. The training and optimization of deep learning algorithms, such as convolutional neural networks (CNN) and long short-term memory (LSTM) networks, can include millions of parameters and substantial processing resources [52].

In contrast, time-tested techniques OPF and droop control are both computationally light and amenable to real-time implementations. Simple proportional-integral (PI) controllers can be used for droop control, while linear programming techniques can be used to solve OPF.

The development of more effective machine learning algorithms for use in power system control applications has been the focus of current research. In a neural network, for instance, hardware acceleration may speeds up computation while pruning techniques can minimize the number of parameters.

# 6. Conclusion & Future Scope

This section concludes the whole thesis discussing different aspects of study and main results obtained from the experiments and simulations. Moreover, some light has been thrown on possible future work that could continue this research.

The dissertation includes basics of Machine learning algorithms, its applications in power systems focusing on reactive power prediction and implementations of ML algorithms for time series prediction.
Focusing on reactive power prediction, this thesis covers Introduction, related literature work, power flow analysis of CIGRE networks and implementing three Algorithms i.e. CNN (Convolution Neural Network), LSTM (Long Short-Term Memory) and Neural Prophet.

Power flow analysis is done to see the impact of reactive power on SDN and from the results it can be observed that reactive power play's huge role in voltage regulation in power systems. Then optimal power flow analysis is carried out to get the appropriate data for training ML algorithms because OPF is one of the best conventional approaches used in power industry. Finally, ML algorithms are trained on this data and then predicted output is compared with the standard data. It can be observed that ML can be a very useful tool in predicting reactive power and it can be a very good alternative to traditional approaches like droop control or Optimal power flow analysis which are highly grid models dependent and are not that much flexible. Among all the algorithms implemented LSTM performs the best, although neural prophet was best suited for this application but due to less available data LSTM outperforms. Though ML has vast scope of applications in different fields, but this study provides solid grounds that ML is worth trying to solve problems related to power industry and can be very effective in prediction of reactive power in smart distribution network. The work contributes to the prediction of reactive power support in smart distribution networks utilizing machine learning. This will assist the operators to take timely decisions in order to avoid any instability in the network.

Implementing control system on top of predicted output by ML could be a good step ahead for future work. Furthermore, if we use larger portions of data there is a bright possibility that the neural prophet algorithm will outperform all other algorithms increasing accuracy and efficiency.

# References

[1] J. &. M. R. &. M. T. &. C. J. Carbonell, "Machine Learning: A Historical And Methodological Analysis," *Artificial Intelligence Magazine,* 2002.

[2] Dridi, Salim. (2021). Unsupervised Learning - A Systematic Literature Review. .

[3] M. Mojumder, "Power System Stability Analysis using Neural Network," *IEEE,* 2023.

[4] S. Massoud Amin and B. F. Wollenberg, "Toward a smart grid: power delivery for the 21st century," in IEEE Power and Energy Magazine, vol. 3, no. 5, pp. 34-41, Sept.-Oct. 2005, doi: 10.1109/MPAE.2005.1507024.

[5] K. Moslehi, R. Kumar, Vision for a self-healing power grid, in: ABB Review, No.4, 2006.

[6] S. Amann, S. Proksch, S. Nadi and M. Mezini, "A Study of Visual Studio Usage in Practice," 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), Osaka, Japan, 2016, pp. 124-134, doi: 10.1109/SANER.2016.39.

[7] V. Vapnik. Estimation of Dependences Based on Empirical Data. Springer Verlag, 1982

[8] S. ALSHATTNAWI, "EVALUATION OF DEEP LEARNING AND MACHINE LEARNING ALGORITHMS IN INTRUSION DETECTION SYSTEMS," *Journal of Theoretical and Applied Information Technology,* vol. 101, 2023.

[9] Tomin, Nikita & Kurbatsky, V. & Panasetsky, D.A. & Sidorov, Denis & Zhukov, Aleksei. (2018). Voltage/VAR Control and Optimization: AI approach. IFAC-PapersOnLine. 51. 103-108. 10.1016/j.ifacol.2018.11.685.

[10] W. X. G. S. Jianhong Wang, "Multi-Agent Reinforcement Learning for Active voltage control," *arXiv:2110.14300v5 [cs.LG],* 2022.

[11] Tomin, Nikita & Kurbatsky, Victor & Reutsky, Ivan. (2019). A Hybrid Intelligent Technique for Voltage/VAR Control in Power Systems. IET Generation, Transmission & Distribution. 13. 10.1049/iet-gtd.2019.0214.

[12] Kazmi, Syed Ali & Shahzad, Muhammad & Khan, Akif & Shin, Dong. (2017). Smart Distribution Networks: A Review of Modern Distribution Concepts from a Planning Perspective. Energies. 10. 10.3390/en10040501.

[13] Pessoa, Tiago & Medeiros, Raul & Nepomuceno, Thiago & Bian, Gui-Bin & Albuquerque, V.H.C. & Filho, Pedro Pedrosa. (2018). Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications. IEEE Access. PP. 1-1. 10.1109/ACCESS.2018.2874767.

[14] Comparison of voltage control methods in distribution systems using Q-V based PI and droop controls of solar inverters," 2017 IEEE Power & Energy Society General Meeting, Chicago, IL, pp. 1–5, by S. Nowak, M. S. Metcalfe, W. Eberle, and L. Wang (2017)..

[15] Wu, Tsai-Fu & Huang, Yen-Hsiang & Lin, Ting-Hung. (2018). Multi-Function High-Power Converters for Smart-Grid Applications. E3S Web of Conferences. 69. 01007. 10.1051/e3sconf/20186901007.

[16] Wagle, R., Sharma, P., Sharma, C., Amin, M., Rueda, J.L., Gonzalez-Longatt, F.: Optimal power flow-based reactive power control in smart distribution network using real-time cyber-physical co-simulation framework. *IET Gener. Transm. Distrib.* 00, 1– 14 (2023).

[17] Evgeniou, Theodoros & Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-7_12.

[18] Webster, J.G., Dragičević, T., Meng, L., Blaabjerg, F., and Li, Y. (2020). Control of Power Converters in ac and dc Micro grids. In Wiley Encyclopedia of Electrical and Electronics Engineering, J.G. Webster
(Ed.)

[19] Palizban O, Kauhaniemi K, Guerrero JM. "Microgrids in active network management-Part I: Hierarchical control, energy storage, virtual power plants, and market participation," Renewable and Sustainable Energy Reviews, 36, pp.428-439, 2014. https://doi.org/10.1016/j.rser.2014.01.016

[20] Rocabert J, Luna A, Blaabjerg F, Rodríguez P. Control of Power Converters in AC Microgrids. IEEE Transactions on Power Electronics, 27

[21] D. W. a. M. S. H. Peng, "Comparative study of power droop control and current droop control in DC microgrid," in *The 11th IET International Conference on Advances in Power System Control Operation and Management (APSCOM 2018)*, Hong Kong, China, 2018.

[22] Tzanis, George & Katakis, Ioannis & Partalas, Ioannis & Vlahavas, I.. (2006). Modern Applications of Machine Learning. .

[23] A. G. A. K. Javad Ansari, "Multi-agent systems for reactive power control in smart grids," *Electrical Power and Energy Systems*, pp. 411-425, 2016.

[24] D. Błaszczok, T. Trawiński, M. Szczygieł and M. Rybarz, "Forecasting of Reactive Power Consumption with the Use of Artificial Neural Networks. Electronics," vol. 11, 2022.

[25] Xu Xiao-hui, Zhai Chang-guo, Chen Li-juan, Chen Dong-lei and Yang Yong-biao, "Research on smart distribution network system architecture," IEEE PES Innovative Smart Grid Technologies, Tianjin, China, 2012, pp. 1-4, doi: 10.1109/ISGT-Asia.2012.6303231.

[26] G. S. C. &. Y. P. Jyothi, "Reactive Power Support from End Customer Equipment," *Technol Econ Smart Grids Sustain Energy*, vol. 17, no. 2, 2017.

[27] Jovel, J., & Greiner, R. (2021). An Introduction to Machine Learning Approaches for Biomedical Research. Frontiers in Medicine, *8*. https://doi.org/10.3389/fmed.2021.771607

[28] F. Z. Peng, Y. W. Li, and L. M. Tolbert, Control and protection of power electronics interfaced distributed generation systems in a customer-driven microgrid, in 2009 IEEE Power Energy Society General Meeting, 2009; pp 1–8.

[29] Alrajhi Alsiraji, H., & El-Shatshat, R. (2018). Serious Operation Issues and Challenges Related to Multiple Interlinking Converters Interfacing a Hybrid AC/DC Microgrid. 2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE)

[30] Krechel, T., Sanchez, F., Gonzalez-Longatt, F., Chamorro, H.R., Rueda, J.L.: A transmission system friendly micro-grid: Optimizing active power losses. In: 2019 IEEE Milan PowerTech, pp. 1–6 (2019)

[31] W. X. G. S. Jianhong Wang, "Multi-Agent Reinforcement Learning for Active Voltage Control on Power Distribution Networks," *arXiv:2110.14300v5 [cs.LG]* , 2022.

[32] Aminifar, Farrokh & Abedini, Moein & Amraee, Turaj & Jafarian, Peyman & Samimi, Mohammad Hamed & Shahidehpour, M.. (2021). A review of power system protection and asset management with machine learning techniques. Energy Systems. 13. 1-38. 10.1007/s12667-021-00448-6.

[33] Ander Ordono, Eneko Unamuno, Jon Andoni Barrena, Julen Paniagua, "Interlinking converters and their contribution to primary regulation: a review," International Journal of Electrical Power & Energy Systems, 111, pp.44-57, 2019. https://doi.org/10.1016/j.ijepes.2019.03.057

[34] Joorabian, Mahmood & Hooshmand, Rahmat-Allah. (2005). APPLICATION OF ARTIFICIAL NEURAL NETWORKS IN VOLTAGE AND REACTIVE POWER CONTROL. 12.

[35] Pettersen, D., Melfald, E., Chowdhury, A., Acosta, M.N., Gonzalez-Longatt, F., Topic, D.: TSO-DSO performance considering volt-var control at smart-inverters: Case of vestfold and telemark in Norway.

[36] CIGRE Task Force C6.04.02 (2011) Benchmark Systems for Network Integration of Renewable Energy Resources, Version 7

[37] Ali, Jehad & Khan, Rehanullah & Ahmad, Nasir & Maqsood, Imran. (2012). Random Forests and Decision Trees. International Journal of Computer Science Issues(IJCSI). 9.

[38] Aziz, NorShakirah & Akhir, Emelia & Aziz, Associate Professor Dr Izzatdin & Jaafar, Jafreezal & Hasan, Mohd Hilmi & Abas, Ahmad. (2020). A Study on Gradient Boosting Algorithms for Development of AI Monitoring and Prediction Systems. 11-16. 10.1109/ICCI51257.2020.9247843.

[39] Sarvepalli, Sarat Kumar. (2015). Deep Learning in Neural Networks: The science behind an Artificial Brain. 10.13140/RG.2.2.22512.71682.

[40] Deng, Li. 'Deep Learning: Methods And Applications'. Foundations and Trends in Signal Processing 7.3-4 (2013): 197-387. Web.

[41] Lipton, Zachary. (2015). A Critical Review of Recurrent Neural Networks for Sequence Learning.

[42] O'Shea, Keiron & Nash, Ryan. (2015). An Introduction to Convolutional Neural Networks. ArXiv e-prints.

[43] Yuan, F.-N & Zhang, L. & Shi, J.-T & Xia, X. & Li, G.. (2019). Theories and Applications of Auto-Encoder Neural Networks: A Literature Survey. Jisuanji Xuebao/Chinese Journal of Computers. 42. 203-230. 10.11897/SP.J.1016.2019.00203.

[44] Fang, Jingying & Zhang, Xiangyun. (2019). Research on Power System Relay Protection Method Based on Machine Learning Algorithm. E3S Web of Conferences. 136. 02012. 10.1051/e3sconf/201913602012.

[45] Panda, Mrutyunjaya & Abraham, Ajith & Das, Swagatam & Patra, Manas. (2011). Network intrusion detection system: A machine learning approach.

Intelligent Decision Technologies (IDT) Journal, IOS Press. 5. 347-356. 10.3233/IDT-2011-0117.

[46] L. Thurner et al., "Pandapower—An Open-Source Python Tool for Convenient Modeling, Analysis, and Optimization of Electric Power Systems," in IEEE Transactions on Power Systems, vol. 33, no. 6, pp. 6510-6521, Nov. 2018, doi: 10.1109/TPWRS.2018.2829021.

[47] Wang X, Huang T, Zhu K, Zhao X. LSTM-Based Broad Learning System for Remaining Useful Life Prediction. *Mathematics*. 2022; 10(12):2066.

[48] Shohan MJA, Faruque MO, Foo SY. Forecasting of Electric Load Using a Hybrid LSTM-Neural Prophet Model. *Energies*. 2022; 15(6):2158.

[49] Rundo,. (2019). Deep LSTM with Reinforcement Learning Layer for Financial Trend Prediction in FX High Frequency Trading Systems. Applied Sciences. 9. 4460. 10.3390/app9204460.

[50] Alla, Sri & Athota, Kavitha. (2022). Brain Tumor Detection Using Transfer Learning in Deep Learning. Indian Journal Of Science And Technology. 15. 2093-2102. 10.17485/IJST/v15i40.1307.

[51] R. Wagle, P. Sharma, C. Sharma, M. Amin, and F. Gonzalez-Longatt, “Real-Time Volt-Var Control of Grid Forming Converters in DER-enriched Distribution Network,” Front. Energy Res., no. January, pp. 1–18, 2023.

[52] Text generated by ChatGPT 2023, OpenAI, https://chat.openai.com/chat. Edited for style and content

[53] Zhou, Dingyi & Zuo, Xiaoqing & Zhao, Zhifang. (2022). Constructing a Large-Scale Urban Land Subsidence Prediction Method Based on Neural Network Algorithm from the Perspective of Multiple Factors. Remote Sensing. 14. 1803. 10.3390/rs14081803.

## Appendix 1: DER’s rating [16, 30]

Rating of all DER’s considered in this research

| Name Of  DER | P ( KW) | Q (KVAR) |
|---|---|---|
| DER1 | 690.00 | 303.00 |
| DER2 | 690.00 | 303.00 |
| DER3 | 690.00 | 303.00 |
| DER4 | 680.00 | 300.00 |
| DER5 | 680.00 | 300.00 |
| DER6 | 680.00 | 300.00 |
| DER7 | 680.00 | 300.00 |
| DER8 | 680.00 | 300.00 |
| DER9 | 740.00 | 325.00 |
| DER10 | 740.00 | 325.00 |
| DER 11 | 740.00 | 325.00 |
| DER 12 | 740.00 | 325.00 |
| DER 13 | 740.00 | 325.00 |

# Appendix 2: OPF Code

In this appendix Important sectors of Code for Optimal Power Flow analysis is shown which is summarised version of actual code

```
# Taking data for 13 PVs
OPF_PV_P_cigre = np.transpose(OPF_PV_P_cigre)[:,:13]
PV_Q_cigre = np.transpose(PV_Q_cigre)[:,:13]
# Importing Few PandaPower Libraries for Making CIGRE Network other Functions like OPF
import pandapower
import pandapower.networks
import pandapower.topology
import pandapower.plotting
import pandapower.converter
import pandapower.estimation
import pandapower.test
from pandapower.plotting.plotly import simple_plotly
from pandapower.networks import mv_oberrhein
import pandapower.networks as pn
import pandapower as pp
############### Reading all the load profiles directory paths from single phase load data
directory ##############
import numpy as np
import pandas as pd
import glob

import pandapower.control as control
import pandapower.networks as nw
import pandapower.timeseries as timeseries
from pandapower.timeseries.data_sources.frame_data import DFData
Loading the p_mw and q_var data from the excel data for loads
load_data_P_values_df          =          pd.read_csv("{}single          phase          load
data/load_profile_1_PQ.csv".format(directory_path))
load_data_P_values_df.rename(columns={'P': 'P_1'}, inplace=True)
load_data_P_values_df = load_data_P_values_df[['B1','P_1']]
###Changing the Power Factor Values of all loads ###
updated_pf_values = [0.98, 0.97, 0.97, 0.97, 0.97, 0.97, 0.97, 0.97, 0.98, 0.97, 0.95, 0.85,
0.85, 0.85, 0.85, 0.95, 0.85,0.85]
for i, tmp_pf in enumerate(updated_pf_values):
  net.load.iat[i,7] = tmp_pf
################ Setting Transformer Parameters ##############
net.trafo.pfe_kw = 14
net.trafo.i0_percent = 0.07
net.trafo.shift_degree = 0
net.trafo.tap_neutral = 0
net.trafo.vkr_percent= 0.41
net.trafo.tap_min  = 0.9
net.trafo.tap_max= 1.1
net.trafo.tap_step_percent= 1.5
net.trafo.tap_step_degree= 0
net.trafo.tap_side  = "hv"
net.trafo.parallel    = 1
Net.trafo
########## Runing the OPF One time for the Values set for the complete network
net.line["max_loading_percent"] = 70
pp.runopp(net, delta=1e-16)
# create the data source from sgen active and reactive power dataframe and defining Constant
Controller ##############
df_PV_P = pd.DataFrame(OPF_PV_P_cigre, columns = ['PV_1','PV_2','PV_3', 'PV_4','PV_5','PV_6',
'PV_7','PV_8','PV_9','PV_10', 'PV_11','PV_12','PV_13' ]
ds_PV_active_power = DFData(df_PV_P)
PV_P_prof_names = ['PV_1','PV_2','PV_3', 'PV_4','PV_5','PV_6', 'PV_7','PV_8','PV_9','PV_10',
'PV_11','PV_12','PV_13' ]
]
df_PV_Q  =  pd.DataFrame(PV_Q_cigre,  columns  =  ['PV_1','PV_2','PV_3',  'PV_4','PV_5','PV_6',
'PV_7','PV_8','PV_9','PV_10', 'PV_11','PV_12','PV_13' ]
])
ds_PV_reactive_power = DFData(df_PV_Q)
PV_Q_prof_names  =['PV_1','PV_2','PV_3',  'PV_4','PV_5','PV_6',  'PV_7','PV_8','PV_9','PV_10',
'PV_11','PV_12','PV_13' ]
# starting the timeseries simulation for one day -> 288 5 min values.
timeseries.run_timeseries(net, verbose=True, continue_on_divergence=True)
```

# Appendix 3: CNN Code

This appendix shows main sections of code for implementing CNN Algorithm

```
# Importing data for PV's obtained after running OPF
import numpy as np
OPF_Bus_Vmag_cigre = np.load("Cigre_data_revised/OPF_Bus_Vmag_cigre.npy")
OPF_PV_P_cigre = np.load("Cigre_data_revised/OPF_PV_P_cigre.npy")
PV_Q_cigre = np.load("Cigre_data_revised/PV_Q_cigre.npy")
print  ("Showing  the  shape  of  Revised  CIGRE  Data  Provided:  ",  OPF_Bus_Vmag_cigre.shape,
OPF_PV_P_cigre.shape, PV_Q_cigre.shape )
#################### Installing Machine Learning Libraries ###################
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Flatten, Dense, Dropout, BatchNormalization
from tensorflow.keras.layers import Conv2D, MaxPool2D
from tensorflow.keras.optimizers import Adam
print(tf.__version__)
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
!pip install pydrive
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
Loading and arranging the p_mw and QVar data for loads ##
load_data_P_values_df           =           pd.read_csv("{}single           phase           load
data/load_profile_1_PQ.csv".format(directory_path))
load_data_P_values_df.rename(columns={'P': 'P_1'}, inplace=True)
# load_data_P_values_df = load_data_P_values_df[['B1','P_1']]
load_data_P_values_df = load_data_P_values_df[['P_1']]
# for i in range (total_loads-1):
for i in range (12):
  # load your timeseries from a file (here csv file)
  df_tmp  =  pd.read_csv("{}single  phase  load  data/load_profile_{}_PQ.csv".format(directory_path,
(i+2)))
  df_tmp.rename(columns={'P': 'P_{}'.format(i+2)}, inplace=True)
  # Place the DataFrames side by side
  tmp_horizontal_stack = pd.concat([load_data_P_values_df, df_tmp[['P_{}'.format(i+2)]]], axis=1)
  load_data_P_values_df = tmp_horizontal_stack
load_data_Q_values_df           =           pd.read_csv("{}single           phase           load
data/load_profile_1_PQ.csv".format(directory_path))
load_data_Q_values_df.rename(columns={'Q': 'Q_1'}, inplace=True)
# load_data_Q_values_df = load_data_Q_values_df[['B1','Q_1']]
load_data_Q_values_df = load_data_Q_values_df[['Q_1']]
# for i in range (total_loads-1):
```

```
for i in range (12):
  # load your timeseries from a file (here csv file)
  df_tmp  =  pd.read_csv("{}single  phase  load  data/load_profile_{}_PQ.csv".format(directory_path,
(i+2)))
  df_tmp.rename(columns={'Q': 'Q_{}'.format(i+2)}, inplace=True)
  # Place the DataFrames side by side
  tmp_horizontal_stack = pd.concat([load_data_Q_values_df, df_tmp[['Q_{}'.format(i+2)]]], axis=1)
  load_data_Q_values_df = tmp_horizontal_stack
##Concatenating all inputs into 1 Array i.e. Loads Active Power, Loads Reactive and PVs Active Power
load_data_P_Q_values = pd.concat([load_data_P_values_df,load_data_Q_values_df], axis=1)
load_P_Q_and_PV_P= pd.DataFrame(np.column_stack([load_data_P_Q_values, OPF_PV_P_cigre]))
######## Showing Concatenated Input ##################
load_P_Q_and_PV_P
########### Scaling the input values by subtrating its means and with a specific standard deviation,
So, that the ML Model can learn Easily #################
scaler_input = StandardScaler()
scaled_load_P_Q_and_PV_P = scaler_input.fit_transform(load_P_Q_and_PV_P)
######### Showing scaled input Values #################
print(scaled_load_P_Q_and_PV_P.shape, scaled_load_P_Q_and_PV_P)
print("Showing the mean of scaled values:", scaler_input.mean_ )
########### Scaling the output values by subtrating its means and with a specific standard deviation,
So, that the ML Model can Converge Easily #################
scaler_output = StandardScaler()
scaled_PV_Q_cigre = scaler_output.fit_transform(PV_Q_cigre)
########### Showing Scaled Output Values ###################
print(scaled_PV_Q_cigre.shape, scaled_PV_Q_cigre)
print("Showing the mean of scaled Q values:", scaler_output.mean_ )
######################## Spliting the Data into Training and Testing Sets ##################
X_train,  X_test,  y_train,  y_test  =  train_test_split(scaled_load_P_Q_and_PV_P,  scaled_PV_Q_cigre,
test_size = 0.2)
############################ Initializing MLP Model Architecture ########################
model = Sequential()
model.add(Dense(32, activation = 'sigmoid', input_shape = X_train[0].shape))
model.add(Dense(26, activation = 'sigmoid'))
# model.add(Dropout(0.5))
model.add(Dense(18, activation = 'sigmoid'))
model.add(Dense(13))
## Showing the MLP Model Architecture Summary
print(model.summary())
# Training the MLP Model with the Training and Validation Data #
history = model.fit(X_train, y_train, epochs = 70, validation_data= (X_test, y_test), verbose=1)
################## Finding the Predictions from Saved Model ##################
from mlxtend.plotting import plot_confusion_matrix
from sklearn.metrics import confusion_matrix
from tensorflow.keras.utils import to_categorical
y_pred = model.predict(scaled_load_P_Q_and_PV_P)
# Inverse Scaling the Predicted Reactive Power By MLP and Given Reactive power for training #########
PV_Q_cigre_after = scaler_output.inverse_transform(scaled_PV_Q_cigre)
```

# Appendix 4: Neural Prophet Code

```
!pip install neuralprophet
 # Importing required libraries
import neuralprophet
import numpy as np
import pandas as pd
Merges multiple data frames
merged_df = pd.read_csv('single phase load data/load_profile_1_PQ.csv')

for i in range(2, 14):
    filename = f"single phase load data/load_profile_{i}_PQ.csv"
    df = pd.read_csv(filename)

    # Concatenate the DataFrame horizontally
    merged_df = pd.concat([merged_df, df[['P', 'Q']]], axis=1)

# Data preprocessing
merged_df = merged_df.rename({"B1": "timestamp"}, axis=True)
base_date = pd.to_timedelta('00:00:00')
merged_df['timestamp'] = base_date + pd.to_timedelta(merged_df['timestamp'], unit='minute')

timedelta_col = pd.to_timedelta(merged_df['timestamp'], unit='m')
merged_df['timestamp'] = timedelta_col.apply(lambda x: str(x).split()[-1])
def outlier_detector(data):
    # Define the window size for rolling statistics
    window_size = 12

    # Calculate rolling mean and standard deviation for each column
    rolling_mean = data.rolling(window_size).mean()
    rolling_std = data.rolling(window_size).std()

    # Calculate z-scores for each data point based on rolling statistics
    z_scores = np.abs((data - rolling_mean) / rolling_std)

    # Define the threshold for outliers
    threshold = 3

    # Identify outliers
    outliers = np.where(z_scores > threshold)

    # Print the outliers
    print("Outliers:")
    print(data.iloc[outliers])
# Merging data into single frame
df1_cols = merged_df.iloc[:, :3]
df2_cols = PV_P_df.iloc[:, :1].rename(columns={'0': 'PV_P'})
```

```python
df2_cols = df2_cols.rename(columns={0: 'PV_P'})
df3_cols = PV_Q.iloc[:, :1].rename(columns={'0': 'PV_Q'})
df3_cols = df3_cols.rename(columns={0: 'y'})
PV1_df = pd.concat([df1_cols, df2_cols, df3_cols], axis=1)
# Training data
train_data = merged_df.iloc[:250, 1:]
test_data = merged_df.iloc[250:, 1:]

# Evaluation Functions
def evaluate(y_true, y_pred):
    mae = round(mean_absolute_error(y_true, y_pred), 3)
    mse = round(mean_squared_error(y_true, y_pred), 3)
    rmse = round(np.sqrt(mse),3)
    mape = round(np.mean(np.abs((y_true - y_pred) / y_true)) * 100,3)
    smape = round(np.mean(2 * np.abs(y_true - y_pred) / (np.abs(y_true) + np.abs(y_pred))) * 100,3)

    return {"MAE": mae, "MSE": mse, "RMSE": rmse, "MAPE": mape, "SMAPE": smape}
from neuralprophet import NeuralProphet

# Load the data
data = merged_df.iloc[:, :3]

# Format the data
data = data.rename(columns={"timestamp": "ds", "Q_1": "y"})
data["ds"] = pd.to_datetime(data["ds"])
# data = data.set_index("ds")

# Create the model
model_np_1 = NeuralProphet(
    n_changepoints=100,
    changepoints_range=0.8,
    n_lags=10,
    yearly_seasonality=False,
    weekly_seasonality=False,
)
model_np_1.add_lagged_regressor('P_1')
# Train the model
metrics = model_np_1.fit(data, freq="5min", metrics=["MSE", "MAE", "RMSE"])
# Make predictions
future = model_np_1.make_future_dataframe(data, periods=24*60//5, n_historic_predictions=len(data))
forecast = model_np_1.predict(future)

# Plot the predictions
fig = model_np_1.plot(forecast)
plt.show()
```

# Appendix 5: LSTM Code

```
import zipfile

# specify the path to the zip file
zip_path = '/content/files.zip'

# specify the path to the directory where you want to extract the contents of the zip file
extract_path = '/content/extracted_files/'

# create a ZipFile object
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    # extract all contents to the specified directory
    zip_ref.extractall(extract_path)
#################### Installing Machine Learning Libraries ####################
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Flatten, Dense, Dropout, BatchNormalization
# from tensorflow.keras.layers import Conv2D, MaxPool2D
from tensorflow.keras.layers import LSTM

from tensorflow.keras.optimizers import Adam
print(tf.__version__)
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
!pip install pydrive
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
############## Reading all the load profiles directory paths from single phase load data directory
##############
import numpy as np
import pandas as pd
import glob

import pandapower.control as control
import pandapower.networks as nw
import pandapower.timeseries as timeseries
from pandapower.timeseries.data_sources.frame_data import DFData

# number of time steps
n_ts = 288

all_loads_directory_path = "/content/extracted_files/single phase load data/"
all_load_profiles_paths = glob.glob('{}/*.csv'.format(all_loads_directory_path))
```

```
total_loads = len(all_load_profiles_paths)
# Load the P values from the first file
load_data_P_values_df      =      pd.read_csv("/content/extracted_files/single      phase      load
data/load_profile_1_PQ.csv")
load_data_P_values_df.rename(columns={'P': 'P_1'}, inplace=True)
load_data_P_values_df = load_data_P_values_df[['P_1']]

# Load the P values from the remaining files
for i in range(12):
    # Load the timeseries data from a CSV file
              df_tmp      =      pd.read_csv("/content/extracted_files/single      phase      load
data/load_profile_{}_PQ.csv".format(i+2))
    df_tmp.rename(columns={'P': 'P_{}'.format(i+2)}, inplace=True)
    # Place the DataFrames side by side
    tmp_horizontal_stack = pd.concat([load_data_P_values_df, df_tmp[['P_{}'.format(i+2)]]], axis=1)
    load_data_P_values_df = tmp_horizontal_stack.copy()
# View the resulting DataFrame
print(load_data_P_values_df.head())
# Load the Q values from the first file
load_data_Q_values_df      =      pd.read_csv("/content/extracted_files/single      phase      load
data/load_profile_1_PQ.csv")
load_data_Q_values_df.rename(columns={'Q': 'Q_1'}, inplace=True)
load_data_Q_values_df = load_data_Q_values_df[['Q_1']]

# Load the Q values from the remaining files
for i in range(12):
    # Load the timeseries data from a CSV file
              df_tmp      =      pd.read_csv("/content/extracted_files/single      phase      load
data/load_profile_{}_PQ.csv".format(i+2))
    df_tmp.rename(columns={'Q': 'Q_{}'.format(i+2)}, inplace=True)
    # Place the DataFrames side by side
    tmp_horizontal_stack = pd.concat([load_data_Q_values_df, df_tmp[['Q_{}'.format(i+2)]]], axis=1)
    load_data_Q_values_df = tmp_horizontal_stack.copy()

# View the resulting DataFrame
print(load_data_Q_values_df.head())
#################### Concatenating all inputs into 1 Array i.e. Loads Active Power, Loads Reactive
Power and PVs Active Power ####################

load_data_P_Q_values = pd.concat([load_data_P_values_df,load_data_Q_values_df], axis=1)

load_P_Q_and_PV_P=  pd.DataFrame(np.column_stack([load_data_P_Q_values,  OPF_PV_P_cigre]))   #  this
should be reactive power right not active ???

######## Showing Concatenated Input ##################
load_P_Q_and_PV_P
########### Scaling the input values by subtrating its means and with a specific standard deviation,
So, that the ML Model can learn Easily #################
```

```
scaler_input = StandardScaler()
scaled_load_P_Q_and_PV_P = scaler_input.fit_transform(load_P_Q_and_PV_P)
######### Showing scaled input Values #################
print(scaled_load_P_Q_and_PV_P.shape, scaled_load_P_Q_and_PV_P)
print("Showing the mean of scaled values:", scaler_input.mean_ ) merged_df = pd.read_csv('single
phase load data/load_profile_1_PQ.csv')
for i in range(2, 14):
    filename = f"single phase load data/load_profile_{i}_PQ.csv"
    df = pd.read_csv(filename)
    # Concatenate the DataFrame horizontally
    merged_df = pd.concat([merged_df, df[['P', 'Q']]], axis=1)
df = merged_df.set_index('timestamp')
fig, axs = plt.subplots(13, 1, figsize=(10, 30))
axs = axs.flatten()
for i in range(13):
    ax = axs[i]
    p_col = 'P_' + str(i+1)
    q_col = 'Q_' + str(i+1)
    ax.plot(df.index, df[p_col], label=p_col)
    ax.plot(df.index, df[q_col], label=q_col)

    tick_values = ax.get_xticks()[::12]
    ax.set_xticks(tick_values)
    ax.tick_params(axis='x', labelrotation=45)
    ax.legend()
    ax.set_title(p_col + ' and ' + q_col)
plt.tight_layout()
plt.show()
############################ Initializing LSTM Model Architecture ########################
model = Sequential()
model.add(Dense(128, activation = 'sigmoid', input_shape = X_train[0].shape))
model.add(Dense(64, activation = 'sigmoid'))
model.add(Dense(32, activation = 'sigmoid'))
# model.add(Dropout(0.5))
model.add(Dense(26, activation = 'sigmoid'))
# model.add(Dropout(0.5))
model.add(Dense(18, activation = 'sigmoid'))
# model.add(Dropout(0.5))
# model.add(Dense(13, activation = 'tanh'))
model.add(Dense(13))
################ Showing the LSTM Model Architecture Summary #################
print(model.summary())
```