A Machine Learning based High-Speed State Estimator for Partially Observed Electric Transmission Systems

by

Harish Chandrasekaran

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved November 2020 by the
Graduate Supervisory Committee:

Anamitra Pal, Chair
Arunabha Sen
Daniel Tylavsky

ARIZONA STATE UNIVERSITY

December 2020

ABSTRACT

The accurate monitoring of the bulk transmission system of the electric power grid by sensors, such as Remote Terminal Units (RTUs) and Phasor Measurement Units (PMUs), is essential for maintaining the reliability of the modern power system. One of the primary objectives of power system monitoring is the identification of the snapshots of the system at regular intervals by performing state estimation using the available measurements from the sensors. The process of state estimation corresponds to the estimation of the complex voltages at all buses of the system. PMU measurements play an important role in this regard, because of the time-synchronized nature of these measurements as well as the faster rates at which they are produced. However, a model-based linear state estimator created using PMU-only data requires complete observability of the system by PMUs for its continuous functioning. The conventional model-based techniques also make certain assumptions in the modeling of the physical system, such as the constant values of the line parameters. The measurement error models in the conventional state estimators are also assumed to follow a Gaussian distribution.

In this research, a data mining technique using Deep Neural Networks (DNNs) is proposed for performing a high-speed, time-synchronized state estimation of the transmission system of the power system. The proposed technique uses historical data to identify the correlation between the measurements and the system states as opposed to directly using the physical model of the system. Therefore, the highlight of the proposed technique is its ability to provide an accurate, fast, time-synchronized estimate of the system states even in the absence of complete system observability by PMUs.

The state estimator is formulated for the IEEE 118-bus system and its reliable performance is demonstrated in the presence of redundant observability, complete observability, and incomplete observability. The robustness of the state estimator is also demonstrated by performing the estimation in presence of Non-Gaussian measurement errors and varying line parameters. The consistency of the DNN state estimator is demonstrated by performing state estimation for an entire day.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to Dr. Anamitra Pal for providing me with an opportunity to pursue research under his guidance as a member of his research group, the Pal Lab. He has guided me at all stages of my research journey and has been a constant source of motivation. I thank all the members of Pal Lab for the help that they have offered me in my master's journey. A special thanks to Behrouz Azimian and Antos Varghese for their valuable contributions to select portions of my research. I would also like to acknowledge the support provided by Pooja Gupta and Reetam Sen Biswas, who often shared their research experiences with me. Finally, I would like to thank my fellow master's students Hashem Albhrani and John Patterson for constantly encouraging me in the pursuit of my master's degree.

I would like to extend my gratitude to Dr. Arunabha Sen and Dr. Daniel Tylavsky for their interest in being part of my thesis committee and helping me in my research with valuable feedback and insights.

My heartful thanks to my parents Chandrasekaran and Bhuvaneswari, for being by pillar of strength and for their continuous emotional and financial support. I thank my sister Harini and all the members of my family for their constant support and encouragement. I am very thankful to my housemates Varoon, Madhumadhavan and Danish for their understanding and support that they have provided during my master's journey. A wholehearted acknowledgement to all my friends, especially from my undergraduate college at NIT Trichy and graduate college at Arizona State University and my well-wishers for their contributions towards my holistic growth.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF TABLES

Table Page

Table Page

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

The rise in the technological advancements of the modern world has been reflected proportionately in the increase in demand for electricity, so much so that the requirement of a continuous and reliable power supply has become *a basic need*. At the same time, the outages in the power system have also seen an unfortunate rise in recent times. For instance, the state of California has been experiencing rolling outages in 2020, the biggest happening in the state since 2001 [1] . These power outages can be avoided by building a resilient power system which encompasses a multitude of sensors for its protection and control. The power system protection and controls required for a resilient power grid can only be achieved by monitoring the system in real-time.

1.1 Power System Monitoring

The technological advancements made in power system monitoring can be dated back to the 1960s with the introduction of the Supervisory Control and Data Acquisition (SCADA) systems. A Remote Terminal Unit (RTU) made using microprocessors, interfaced with the devices in the physical world and converted them into signals that were transferred to the SCADA control center. However, the 1965 blackout that occurred in the North Eastern US and portions of Canada affecting over 30,000,000 customers, amongst other outages, made power engineers realize the need for improved coordination among different utilities for reliable power system operations and planning [2] . Eventually in 1970, Fred Schweppe came up with the idea of power system state estimation [3] , the most important component of power system monitoring. The states of the system were the complex voltages (voltage

magnitudes and voltage angles) of all the buses present in the system. The power system state estimation corresponds to the formulation of a methodology to estimate the states of the system from the available measurements. The estimated states of the system are essential for numerous power system operations and planning applications [4] . Fig 1.1 given below demonstrates the functioning of a modern real-time Energy Management System (EMS).

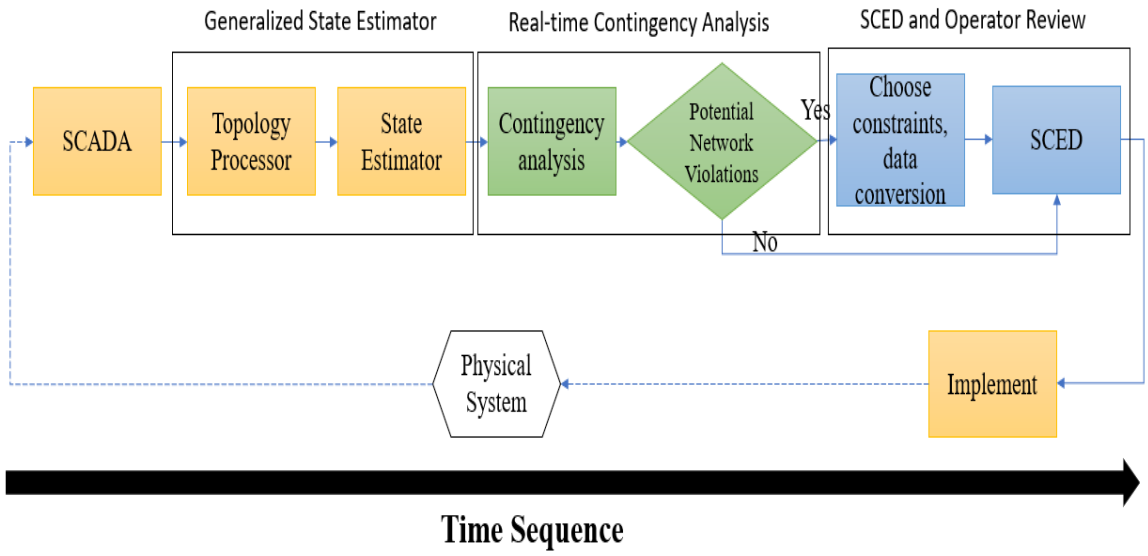

Figure 1.1: Functioning of an EMS [5]

It can be observed that state estimation is the primary step in the operation of an EMS as the estimated system states are utilized for performing Real-Time Contingency Analysis (RTCA) and Security Constrained Economic Dispatch (SCED). RTCA involves checking and calculating violations in different quantities (voltages, power flows) that could potentially cause outages in the power systems. SCED involves identification of the power dispatch of different generators in the system [6] . Both of these applications (RTCA and SCED) rely on the SCADA-based state estimator for their reliable operation. The SCADA

measurements comprise the real and reactive power flows on either side of the branches, real and reactive power injections, and voltage magnitude measurements. However, these measurements are not time stamped and the voltage angles are not measured directly.

A more recent device that has completely revolutionized power system monitoring is the Phasor Measurement Unit (PMU). It is a device that can measure the complex voltages of a bus and the currents in the branches coming out of the bus in a time synchronized manner. The internal architecture of a PMU is shown in Fig 1.2 below.

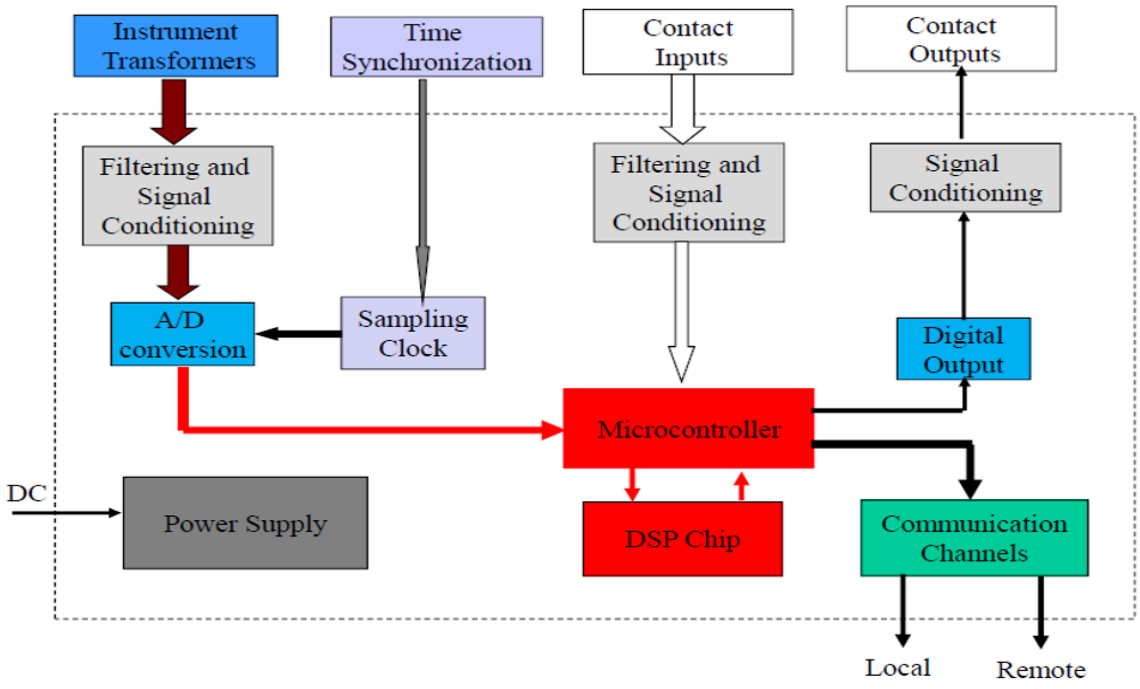

Figure 1.2: Architecture of PMU [7]

Traditionally, a PMU makes use of a Recursive DFT algorithm to calculate the phasor values of the voltage and the current [8] , [9] . The calculated phasor is mathematically defined by:

$$\overline{X^N} = X^{N-1} + \frac{\sqrt{2}}{N}\left[x(N)e^{-j*2\pi N\theta} - x(0)\right] \qquad (1.1)$$

where

$$\overline{X^N} - Calculated\ phasor$$

$$X^{N-1} - Calculated\ value\ of\ previous\ phasor$$

$$N - Sampling\ rate\ of\ the\ ADC\ in\ the\ PMU$$

$$x(N) - Waveform\ of\ the\ current\ sample$$

$$\theta - Phase\ angle\ of\ the\ current\ sample, where\ \theta = \frac{2\pi}{N}$$

The PMU-based synchrophasor technology takes inputs from voltage and current transformers (instrument transformers) which step down the voltages and currents to levels that the PMU's electronic circuits can safely handle [10] . The PMU device consists of an A/D converter that can produce discrete samples and the produced samples are fed to the microcontroller which performs the phasor estimation algorithm (such as recursive DFT). The PMU device also has a sampling clock which ensures that the measurements are time stamped using the Global Positioning System (GPS).

The measurements from the PMU are highly reliable because of the time synchronized nature of the measurements (unlike the RTU measurements) and the high accuracy of the state estimates (PMUs measure the states directly). The PMU measurements are typically available at the rate of 30 samples per second unlike the SCADA measurements which are available once every 2-4 seconds. Therefore, an estimator formulated with PMU measurements as its inputs will be more accurate and can provide more frequent

information about the power system operation by estimating the system states at a much faster rate than the traditional estimators which utilize RTU data.

1.2 Objective of the Work:

The applications of data analytics in the power industry has grown tremendously over the last two years with the advancements in Artificial Intelligence and increased capabilities of handling Big Data [11] , [12] . The biggest source of Big Data in the power systems are the PMUs because of the higher speeds at which they produce outputs. The formulation of state estimation relies on the availability of measurements/data, and hence data analytics can be easily deployed for this application.

A PMU placed at a bus can observe that bus as well as one or more neighboring buses as it can measure the voltage at that bus and currents in one or more branches connected to that bus. Thus, topological observability of the complete system can be obtained by placing PMUs at selected bus locations given by an appropriate PMU placement formulation, such as [13] . There are many model-based linear/hybrid state estimator formulations utilizing the linear dependency of the states with the PMU measurements of the system [14] , [15] . For instance, [14] discusses two methodologies for performing hybrid state estimation using SCADA and PMU measurements. The first methodology involves mixing PMU measurements with traditional SCADA measurements for a non-linear estimation. The second methodology involves a formulation of an additional linear postprocessing step to include the PMU measurements to the output of the traditional non-linear estimator. The linear state estimators such as the one proposed in [16] utilizes only the highly accurate time synchronized PMU measurements for state estimation. Note that a linear state

estimator requires complete system observability of the system by the PMUs for the states to be correctly estimated [17] . However, the synchrophasor infrastructure is expensive and it may be cost prohibitive to ensure complete system observability of all transmission-level substations. It was shown in [18] -[22] that the cost of disrupting a substation to add a PMU is roughly 20 times the cost of the PMU device.

In contrast, the proposed neural network formulation includes identifying the correlation between the measurements and the state estimates by minimizing the expected error in the estimate as opposed to minimizing the expected error in the model, proposed by the model-based state estimators. A deep learning technique can therefore be used for performing an accurate state estimation only making use of highly accurate, reliable, time synchronized measurements, even in the absence of complete system observability by the PMUs.

Moreover, the proposed data mining technique can perform state estimation without making any explicit assumptions regarding the physical model of the system. The model-based approaches involve formulation of a weight matrix, which is a function of standard deviation of the errors. These model-based approaches assume the errors to be Gaussian and work best when this assumption is true. The errors in the PMU have already been analyzed using field data and are demonstrated to be Non-Gaussian [23] . Since, a methodology using a deep neural network tries to minimize the estimation error directly, it does not make any assumption on the error model of the measurements and will be robust against the error in the measurements irrespective of the distribution that they may have.

Thus, the objective of the work is to formulate a deep neural network-based state estimator for the IEEE 118-bus transmission system that can estimate the states using PMU

measurements even if the PMUs do not completely observe the system in the topological sense. The robustness of the state estimator will be then tested by formulating the state estimator in the presence of different error models, namely, Gaussian errors, Gaussian Mixture Model (GMM) errors, and Laplacian errors. The robustness of the state estimator will also be tested in the presence of varying line parameters of the system. The consistency of the approach will be demonstrated by performing state estimation over the entire day.

1.3: Overview of the Thesis:

The rest of the thesis has been organized in the following manner:

Chapter 2 reviews the various traditional methodologies proposed earlier for performing state estimation utilizing the measurements from SCADA systems and PMUs. It also reviews the data mining-based methodologies proposed earlier for state estimation.

Chapter 3 describes the process of data generation that is required for training the neural network. An overview, formulation and working of the deep neural network-based state estimator is also discussed in this chapter. This chapter details the formulation of a linear state estimator as well.

Chapter 4 tests the performance of the proposed state estimator under the following cases: (1) Estimation in the presence of limited number of sensors, (2) Performance against the linear state estimator (3) Robustness of the state estimator. It also demonstrates the performance of the state estimation over an entire day.

Chapter 5 concludes the conducted research and outlines the different directions in which this work can be extended.

CHAPTER 2

LITERATURE REVIEW

The introduction of the state estimators in power systems dates back to the 1970s when Fred Schweppe introduced the idea of estimating the voltage magnitudes and angles of all the buses from the available RTU measurements [24] , [25] . The equation representing the relationship between the measurements Z and the states X are given by:

$$Z = h(X) + e \tag{2.1}$$

A weighted least squares (WLS) approach was implemented to solve this state estimator, which corresponded to minimizing the following function.

$$J(x) = \left(Z - h(X)\right)^T R^{-1} \left(Z - h(X)\right) \tag{2.2}$$

The matrix h relating the measurements and the states are a function of the operating system conditions and it makes the above equation non-linear. The above equation can be solved using an iterative Gauss Newton approach [26] , [27] . The above formulation also involves the computation of the error covariance matrix given by:

$$G(x^k) = H^T(x^k)\, R^{-1}\, H(x^k) \tag{2.3}$$

where $H = \frac{dh(x)}{dx}$. The estimated states at the end of iteration is given by:

$$[G(x^k)]\, \Delta x^{k+1} = H^T(x^k)\, R^{-1} \left(z - h(x^k)\right) \tag{2.4}$$

The $G(x^k)$ known as the Gain matrix was not always invertible and this resulted in difficulties in computation. This led to the introduction of fast decoupled state estimators

[29] . Another issue with the non-linear WLS method was the computation complexity involved in the formulation of the Jacobian matrix, $H(x)$, for large systems. A multi area state estimator [29] - [31] was formulated for larger systems where state estimation was performed in different areas at local level and all the estimated states were combined to perform state estimation at the control center level. This technique was useful in reducing the computation complexities arising in large systems.

2.1 Wide Area Monitoring

The wide area monitoring in power systems done using PMUs have a variety of applications [32] - [57] . PMUs can help provide a more accurate estimation of the system states than the traditional SCADA systems and at a much faster time scale than the SCADA systems. However, the cost of synchrophasor infrastructure is expensive so it is practically difficult to place enough PMUs for observing the entire system [18] - [22] . Several techniques for including the PMU measurements in a mixed WLS state estimator have been proposed in the past. The technique in [58] utilized the voltage angles measured by the PMU to formulate a linear state estimator for the PMU observed area apart from the existing non-linear state estimator. The average of both the estimates were considered to be the final estimate.

A hybrid state estimator consisting of a linear state estimation formulation that uses interpolated results from a WLS state estimator was proposed in [59] . The Jacobian matrix for the linear state estimator, relating the measurements and the states were updated after every SCADA interval. Two popular state estimation techniques incorporating PMUs include: A mixed hybrid state estimator and a two-stage state estimator. The former

methodology includes a mixture of SCADA measurements and PMU measurements in a single linear state estimator. This technique consists of formulating a state rotation matrix using which the SCADA estimated states and the PMU measurements, transformed into rectangular coordinates, can be used to accurately estimate the states of the system in cartesian form [15] , [60] . The latter includes the formulation of a two-stage fusion estimator [61] , [62] . The first stage consists of formulating a linear state estimator and estimating the PMU observed buses in a rectangular form. The rectangular PMU estimated states are then transformed to polar coordinates and sent to a non-linear Weighted least squares-based state estimator. However, both techniques utilize PMU data during the interval that aligns closely with the SCADA data and fails to take advantage of the high-speed rates at which the PMU data are obtained. A technique introduced in [62] , performs multi-stage state estimation. The SCADA estimators and the PMU estimators function independently and a fusion of measurements from both the monitoring devices is performed in a separate stage for the final estimates. The methodology in [63] , [64] introduces a robust dynamic state estimator that makes use of the high-speed data rates of PMU for dynamic monitoring. The methodology utilizes an interpolation matrix obtained using a modified WLS to obtain real time estimates of the PMU unobserved area using the high-speed PMU measurements. A bad data detection technique along with the robustness of the estimates in presence of faults was also considered.

A linear three phase state estimator was proposed in [65] where the state estimation was performed in all three phases using only the synchrophasor measurements. Dominion Energy, a utility in Virginia implemented this three-phase linear estimators, independent from its existing SCADA estimator. However, such an estimator requires observability of

the entire system by PMUs [66] . Similarly, all the methodologies mentioned earlier rely on the system topology information and the system parameter knowledge. Therefore, although PMU measurements significantly increase the accuracy of the estimated states of the observed area, the estimated states in the unobserved area will be dependent on the less accurate SCADA measurements.

2.2 Data Mining Techniques

The data mining techniques aim at discovering relevant patterns in large datasets to extract useful information from the data. The patterns of historical measurements and the corresponding state labels can be used for identifying the underlying relationship between the measurements and the states. The data mining techniques can estimate the states without utilizing the physical model of the network. A methodology demonstrated in [67] , [68] uses a kernel ridge regression to perform state estimation, data cleaning, and bad data analysis using historical data. A power system state estimator using neural networks was developed in [69] , [70] where it was shown that the use of neural networks for state estimation in the presence of SCADA data gave better results than the traditional Kalman filter approach [71] .

A neural network state estimator utilizing the synchrophasor measurements was demonstrated for distribution and transmission systems in [72] and [73] , respectively. However, they did not discuss the formulation and performance of the state estimator under different system conditions (such as non-Gaussian noise and varying line parameters).

CHAPTER 3

STATE ESTIMATOR FORMULATION

Any analysis performed using a supervised learning technique[1] requires a set of features and the known corresponding labels for the features to be fed as inputs. The supervised learning block then tries to learn the unknown mapping existing between the features and the labels using various samples of feature, label pair. Once the supervised learning block learns and identifies the mapping using the various samples that are fed as inputs, it will be capable of predicting the corresponding label for any new set of features. A sample is defined as a set of features, labels data pair. The samples of data supplied for the supervised learning block can be divided into training and validation samples. The training samples, huge in number, are used to identify the mapping between the features and labels and the validation samples are used to authenticate the mapping that the supervised learning block learnt using the training samples. The learnt supervised learning block can then predict the labels when tested for any new set of features.

The features of the neural network, which does supervised learning for the analysis done in this thesis, will be the PMU measurements of the power system and the labels will be the states of the system corresponding to the measurements. Fig 3.1 below demonstrates an overview of this formulation. The green arrow indicates the learning of the neural network by the identification of bidirectional mapping existing between the known samples of measurements, states pair. The yellow arrow indicates the transition to a trained neural network after the completion of training and validation.

[1] Supervised learning is one type of machine learning

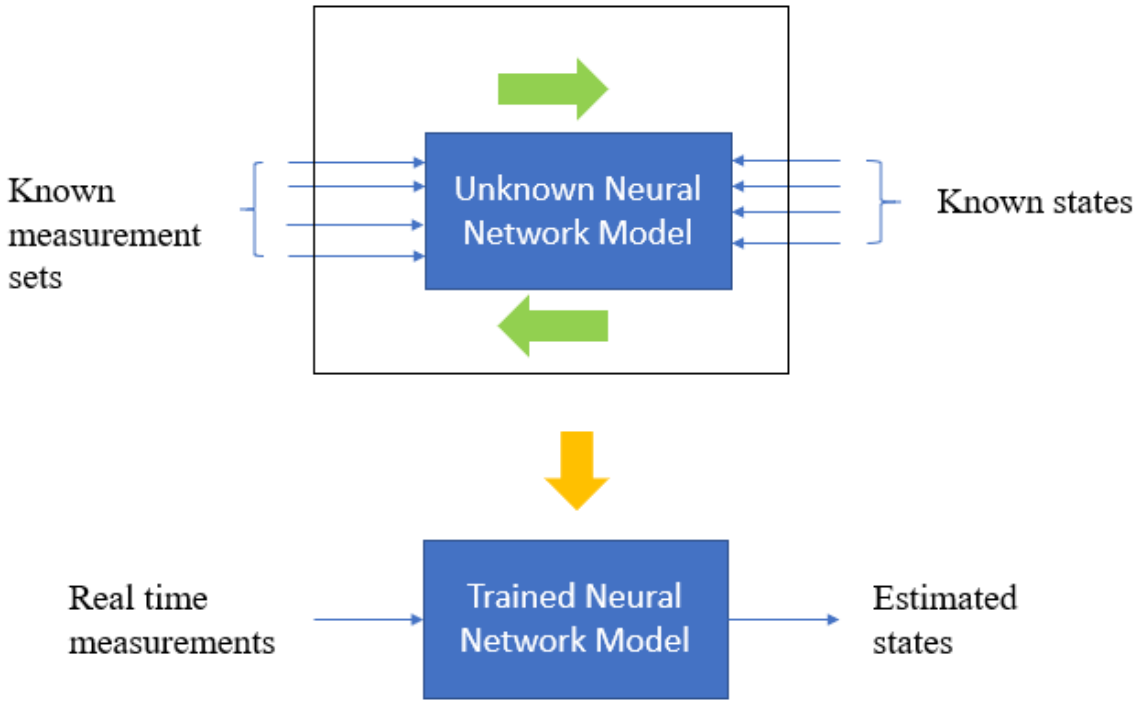

Figure 3.1: Overview of the Neural Network State Estimator

3.1 Data Generation Overview

The first step for performing state estimation will be to generate the data set required for the neural network. The neural network requires various samples of the measurements and the corresponding states for the training and the validation of the network. The measurements to the network will be the fast time scale, high-speed time-synchronized PMU measurements. A PMU can be placed in two different ways [19] , [22] , [74] :

1) A bus-type PMU placed on a bus, measuring the complex voltage of the bus and complex currents in all the branches connected to the bus.
2) A dual used line relay (DULR) PMU placed on a branch, measuring the complex current of the branch and the voltage of the nearby bus to which the branch is connected. A DULR functions both as a relay and a PMU.

The input features to the neural network are the complex voltages and the currents that a PMU can measure. The data required for the training and validation can be obtained from the historical data. Historical PMU data can be directly used as the features of the neural network if they are available. However, it is not highly likely that the “true” states of the system corresponding to the historical measurements will also be available. Also, if any additional PMUs were added to the system, it could result in an absence of historical data for the newly added PMUs. Therefore, synthetic PMU data can be used for the training and the validation of neural networks, while the testing can be done using a stream of available real-time PMU data.

The synthetic PMU data will be generated with the help of available historical load data. The load data is often publicly available for every hour of every day. A load pattern is generated using the available historical load data and a Probability Density Function (PDF) of the load pattern is obtained. Once a PDF of the load pattern is obtained for each load in the system, random load data points are sampled from distribution for the training and validation load data. An ACOPF solution of the sampled points from the load data gives the true voltage values of the system. The true current values of the system are then obtained using the true voltages and the line parameter information. The errors are then added to the true voltage values and current values to obtain the measurements, which are the features of the neural network. The true voltage values generated correspond to the labels of the neural network. The neural network then tries to identify the associated mapping function given by the joint probability distribution of the feature, label pair. Fig 3.2 below summarizes the sequence of steps required for obtaining the data for the neural network from the historical load data.

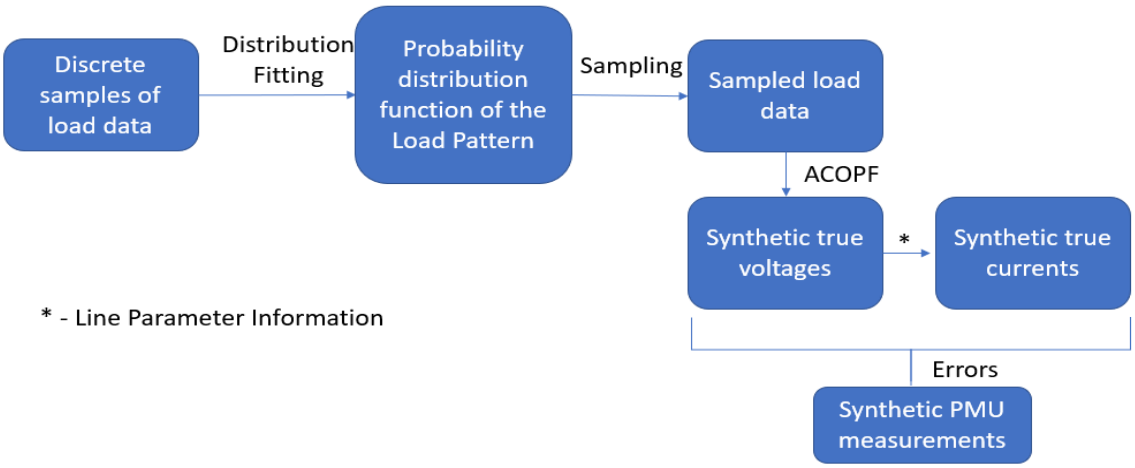

Figure 3.2: Steps for Synthetic PMU Data Generation

3.2 Probability Distribution:

An event refers to the set of outcomes that are possible in an experiment. An event in an experiment can be a continuous event or a discrete event. A probability of an outcome in a discrete event is defined as the number of times an outcome occurred in an event over the total number of possible outcomes. The probability of an outcome X in an event x is defined by:

$$P(x = X) = \frac{Number\ of\ times\ outcome\ X\ happens\ in\ an\ event\ x}{Total\ number\ of\ outcomes\ in\ the\ event\ X} \tag{3.1}$$

However, power system events are not discrete. For example, there can be an extremely large number of hourly load values that are possible. Thus, if a continuous event has a density function $f(x)$, the continuous probability distribution of an outcome x in the range a to b is defined as [75] :

$$P(a \leq x \leq b) = \int_a^b f(x)\,dx \tag{3.2}$$

Thus, the hourly load data points can be fitted into a distribution using a density function and the PDF over a range can be calculated.

The IEEE 118-bus system is chosen to demonstrate the proposed supervised learning-based approach adopted to perform state estimation. The hourly load data for this system is not available online. However, an hourly load data of the synthetic 2000-bus Texas system along with their static power flow information is available online [76] . A load matching approach is adopted which identifies the loads of the synthetic 2000-bus Texas system that resembles closely with the loads of the IEEE 118-bus system. Therefore, by employing this load matching, the available hourly load data of the synthetic 2000-bus Texas system is used to generate hourly load data for the IEEE 118-bus system.

The load demand may be different in a day during different periods of time. For instance, the industrial loads would be very high during the period around afternoon compared to a period during the night. Therefore, an hourly load data during a particular 1-hour interval throughout the year is chosen for obtaining the load distribution and training the neural network for that hour. In the present study, a set of 365 points of load data at 7:00 AM was chosen and neural network was trained and validated for estimating the states during that hour. Fig 3.3 below displays a step plot of the real power of the load over the year and Fig 3.4 below represents a step plot of the reactive power of the load over the year at 7:00 AM.

It can be seen that the plot below represents only a discrete set of data points and the points have to be fitted with a suitable distribution to identify the PDF. The distribution that can be fitted for a discrete set of data points can be classified into two types:

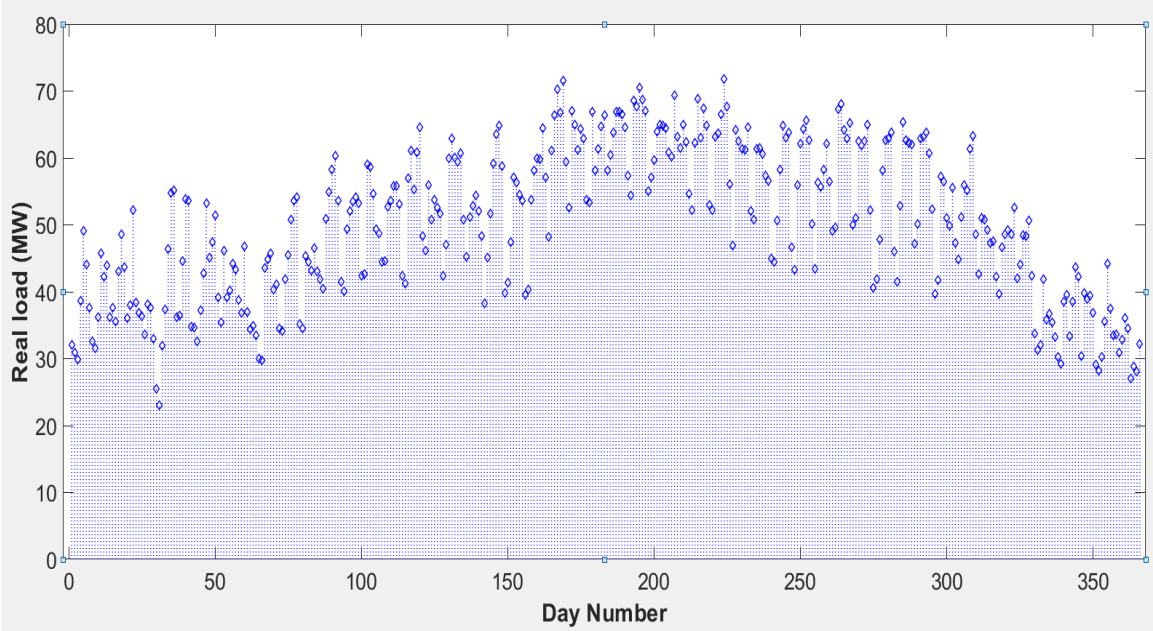

Figure 3.3: Discrete Data Points of Real Load Data at 7:00 AM throughout the Year

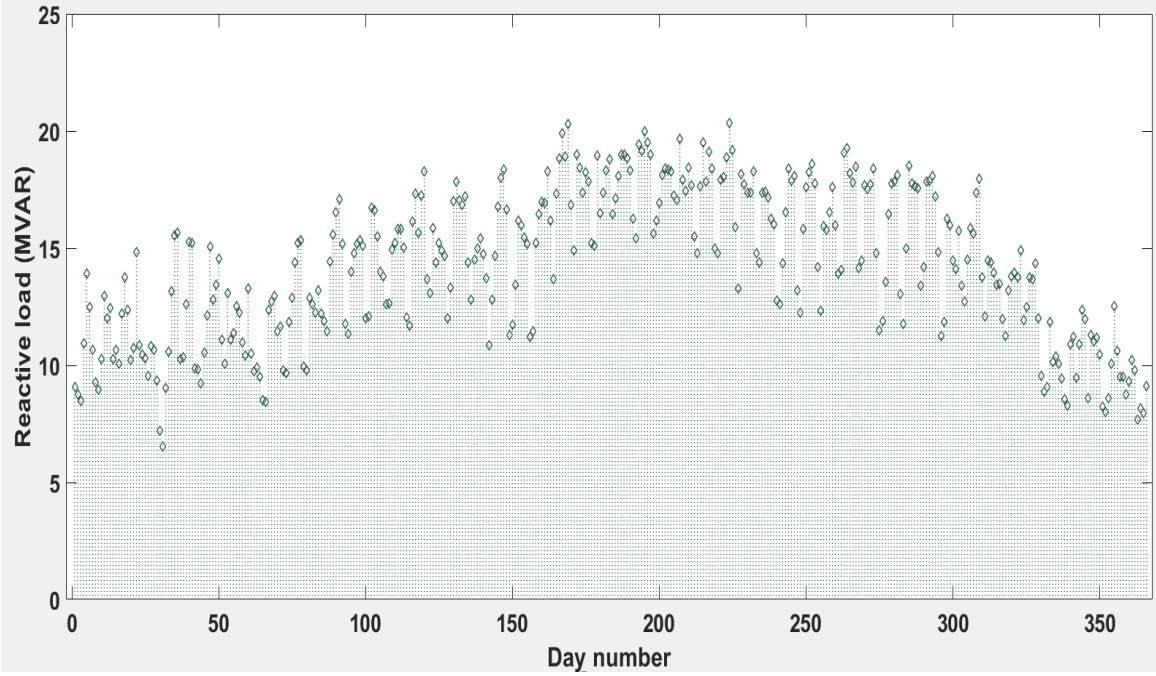

Figure 3.4: Discrete data points of Real Load Data at 7:00 AM throughout the Year

1) Parametric distribution: The probability distribution fitting the set of data points can defined using a set of fixed parameters. Examples of parametric distribution include:

   a) Gaussian (Normal) distribution: The distribution fitting the set of data points is characterized by fixed mean (μ) and standard deviation (σ). The pdf of the Gaussian distribution for a data point x is given by [77] :

$$PDF = \frac{1}{\sigma * 2\pi}\, e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{3.3}$$

   b) Weibull distribution: The distribution fitting the set of data points is characterized by fixed shape parameter (k) and a scale parameter (λ). The pdf of the Weibull distribution for a data point x is given by [78] :

$$PDF = \begin{cases} \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1}\, e^{-\left(x/\lambda\right)^k}, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{3.4}$$

   c) Lognormal distribution: The distribution fitting the set of data points natural logarithm is characterized by fixed mean (μ) and standard deviation (σ).The pdf of the Lognormal distribution for a data point x is given by [79] :

2) Non-parametric distribution: The distribution fitting the set of data points will be a function of a variable parameter. Examples of non-parametric distribution include:

   d) Histogram: A plot that denotes the frequency distribution of the data points over different bins of values. The frequency distribution changes if the width of the bin is varied.

   e) Kernel density estimation: A Kernel density estimator identifies the PDF by a data smoothing approach. The PDF in a kernel estimator is sensitive to the

bandwidth of the estimator. If K_h is a kernel function (Uniform, normal, triangular, epanechikov) and h is the bandwidth of the estimator, the PDF of data point x in the distribution is given by [80] :

$$PDF = \frac{1}{n}\sum_{i=1}^{n} K_h \frac{(x - x_i)}{h} \tag{3.6}$$

where $\{x_1, x_2, x_3, \ldots, x_n\}$ denotes the n discrete data points.

The kernel density estimator is the preferred approach for distribution fitting for discrete load data because of the ability to adjust the distribution to a preferred shape by changing the bandwidth. The hourly real load data points shown above in Fig 3.3 and Fig 3.4 is fitted using both the parametric methods and the non-parametric methods and the result is shown in Fig 3.5 below.

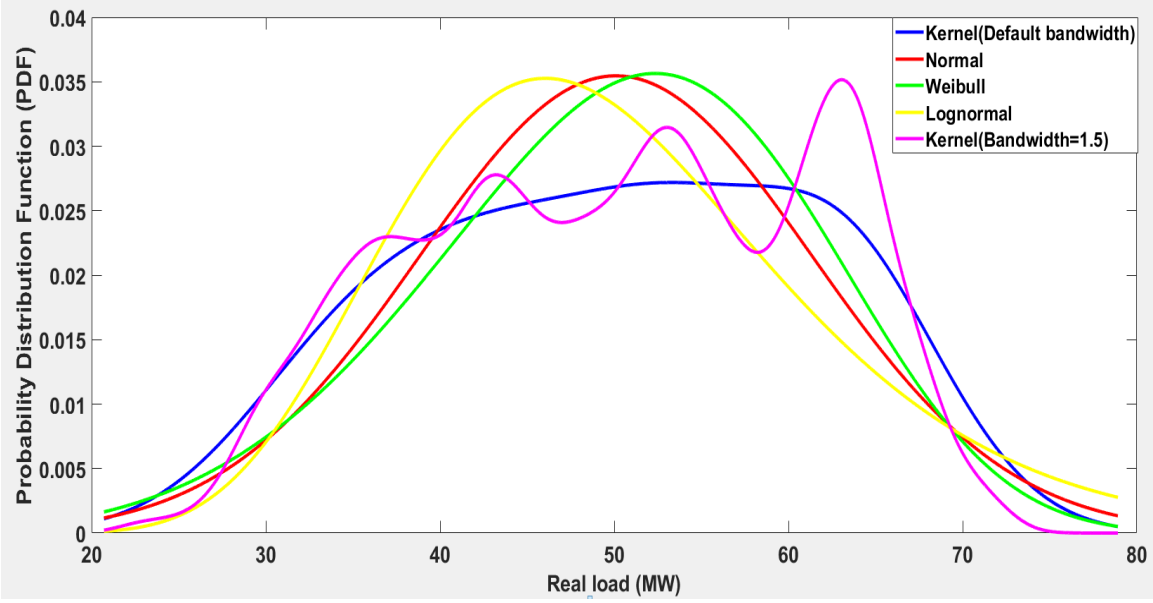

Figure 3.5: PDF of Different Distributions of the Discrete Hourly Load Data

The discrete hourly load data for a particular load of the synthetic 2000-bus Texas system is fitted with Normal, Weibull, Lognormal (parametric) and Kernel (non-parametric) distribution. The parameters of the above fitted distribution are found to be:

Table 3.1: Values of Parameters for Different Fitted Distributions

Distribution	**Parameters**
Kernel (Default bandwidth)	Bandwidth = 4.5844
Normal	μ = 50.0959, σ = 11.0525
Weibull	k = 54.5639, λ = 5.1822
Lognormal	μ = 50.0959, σ = 11.0525
Kernel (Bandwidth = 1.5)	Bandwidth = 1.5

It can be observed that the Kernel density estimator with a bandwidth of 1.5 can fit the discrete real load data points more accurately than a parametric estimator. If the bandwidth is reduced even further, a more accurate representation of the distribution fitting the data points can be obtained.

Now, the goodness of a distribution fitting a set of data points can be ascertained using goodness of fit tests. The goodness of fit tests can again be parametric or non-parametric.

3.3 Parametric Goodness of Fit Test:

The goodness of fit tests returns a parameter p. If the parameter p is greater than a chosen significance level, it implies that the null hypothesis will not be rejected [81] . The rejection

of null hypothesis implies that it is unlikely that the distribution was consistent with the set of data points.

$$\begin{cases} p \geq \alpha, & Null\ hypothesis\ is\ not\ rejected \\ p < \alpha, & Null\ hypothesis\ is\ rejected \end{cases} \tag{3.7}$$

The significance level is conventionally chosen to be 95%. Therefore, if the p value is greater than the 0.5, it indicates that the distribution is a good fit of the given set of data points. A distribution fitting a set of data points better will also have a higher p value. Examples of parametric tests include Chi-square test, Anderson-Darling test, Shapirov-Wilk test, etc. Table 3.2 below demonstrates the calculated p value using the Anderson-Darling test for the 5 distributions mentioned in Fig 3.5. It can be observed that the null hypothesis is rejected for all the parametric distributions.

Table 3.2: p Values of Different Fitted Distributions

Distribution	**p value**
Kernel (Default bandwidth)	0.2542
Normal	0.0245
Weibull	0.0338
Lognormal	0.0041
Kernel (Bandwidth = 1.5)	1.0000

3.4 Non-Parametric Goodness of Fit test:

In a non-parametric good of fit test, the value of the variable parameter which is a measure of the goodness of fit is completely decided by the user. The Dworketsky-Kiefer-

Wolfowitz (DKW) inequality is an example of a non-parametric goodness of fit test [82] . The Cumulative Distribution Function (CDF), $F(x)$, at a data value X is defined as the probability that the value x will be lesser than or equal to X. The CDF $F(x)$ at a data value X for a continuously varying function with PDF $f_n(x)$ is defined as:

$$F(X) = \int_{-\infty}^{X} f(X)\, dx \tag{3.8}$$

An empirical distribution function, $\widehat{F_n}$, is defined as the CDF that puts an equal mass at each data point X in a set of discrete samples with n values.

$$\widehat{F_n}(x) = \frac{\sum_{i=1}^{n} I\ (X_i\ \leq x)}{n} \tag{3.9}$$

where

$$I\ (X_i\ \leq x) = \begin{cases} 1, & if\ X_i\ \leq x \\ 0, & if\ X_i\ \geq x \end{cases} \tag{3.10}$$

Let lower bound $L(x)$ and upper bound $U(x)$ be defined by:

$$L(x) = \max\{\widehat{F_n}(x) - \in_n, \quad 0\} \tag{3.11}$$

$$U(x) = \min\{\widehat{F_n}(x) + \in_n, 1\} \tag{3.12}$$

where

$$\in_n = \sqrt{\frac{1}{2n}\log\left(\frac{2}{\alpha}\right)} \tag{3.13}$$

The α denotes the confidence interval and it is a variable parameter whose desired value must be decided by the user.

A fitted distribution is considered to be a good fit if the following condition is satisfied:

$$P(L(x) \leq F(x) \leq U(x)) \text{ for all } x \geq (1 - \alpha)n \tag{3.14}$$

The confidence interval α is generally considered to be 95%. The above condition implies that the CDF, $F(x)$, can violate the lower and upper bounds for only 5% of the samples.

Fig 3.6 below demonstrates the plot of the CDF, lower bound and upper bound obtained using a KDE of bandwidth 1.5 for the discrete data points shown in Fig 3.3.

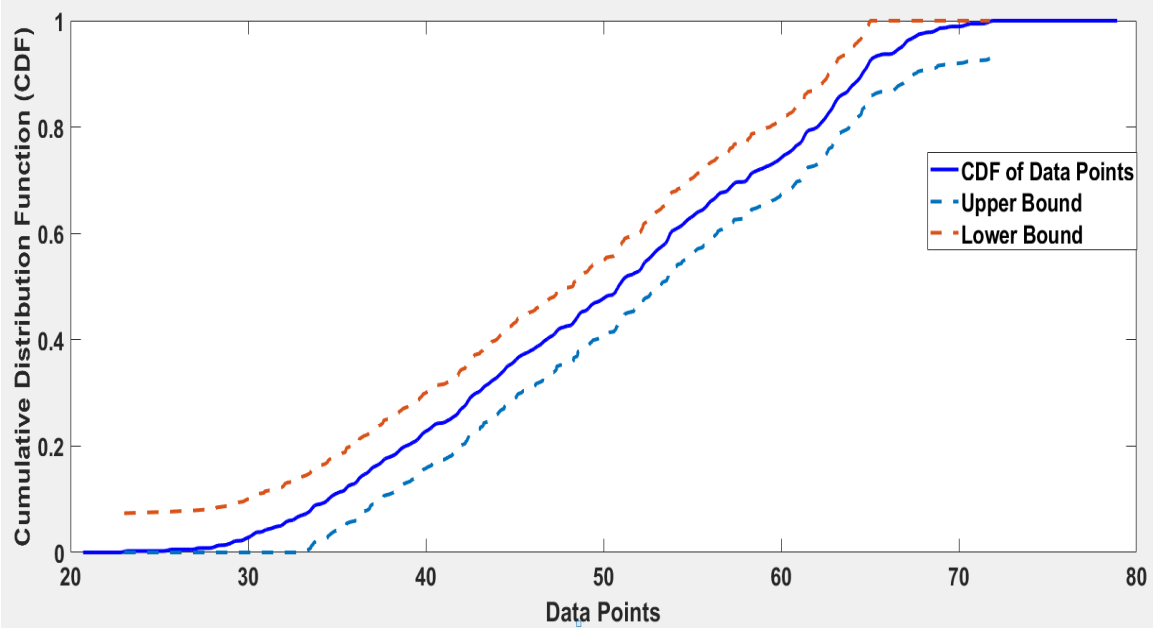

Figure 3.6: Demonstration of DKW Inequality for Sampled Discrete Load Data Points

3.5: Load Matching Methodology:

The discrete data points of both real and imaginary loads of the synthetic Texas 2000-bus system can be fitted into a distribution using the KDE. The default bandwidth of the KDE (given by MATLAB) is chosen to be the initial value. The bandwidth is increased in a loop

in every iteration and is finally stopped if the lower bound conditions of both the parametric and non-parametric tests are satisfied. However, the distribution of loads is required only for the IEEE 118-bus system (since that is the test system) and there is no requirement to identify the distribution of all loads of the synthetic Texas system. Moreover, the loads of the synthetic Texas system should also be chosen in a way that it closely resembles the loads of the IEEE 118-bus system. The load matching is done in the following manner:

Let $Pmean_i$, $Qmean_i$ denote the mean of the discrete load data of bus i obtained for the Texas system. The data for the IEEE 118-bus system is obtained using MATPOWER [83] . Let P_k, Q_k denote the real and reactive load values of the bus k in IEEE 118-bus system given in the MATPOWER. The algorithm for the load matching is described below:

STEP 1: The mean value $Pmean_i$, $Qmean_i$ of the discrete load data points of all the loads of the synthetic Texas system is calculated.

STEP 2: For a load k in the IEEE 118-bus system, the orthogonal difference between the load values of bus k and the non-zero load values of the synthetic Texas system are calculated for all loads of the synthetic Texas system. The difference $diff_{ki}$ is given by:

$$diff_{ki} = \sqrt{(Pmean_i - P_k)^2 + (Qmean_i - Q_k)^2} \tag{3.15}$$

STEP 3: The load distribution of bus i of the IEEE 118-bus system is matched with load k of the synthetic Texas system with the least $diff_{ki}$.

STEP 4: The load pattern for a bus k is obtained by shifting the pdf of the matched load i of the synthetic Texas system by a value of $P_k - Pmean_i$ for real loads and $Q_k - Qmean_i$ for reactive loads.

STEP 5: The required number of random points can now be sampled from the distribution to obtain the discrete load values of the IEEE 118-bus system.

Table 3.3 represents the loads of the synthetic Texas system that are matched with the corresponding load of the IEEE 118-bus system.

Table 3.3: Load Matching Results of IEEE 118-Bus System with 2000-Bus Synthetic Texas System

Load Index	**IEEE 118-Bus system bus number**	**Real load of IEEE 118 bus system (MW)**	**Reactive load of IEEE 118 bus system (MVAR)**	**Texas system bus number**	**Mean Real load of Texas system (MW)**	**Mean Reactive load of Texas system (MVAR)**	**Orthogonal Difference (MW)**
1	1	51	27	1125	54.2957	15.3851	12.07336
2	2	20	9	1262	20.8388	5.9055	3.206109
3	3	39	10	478	38.8108	10.992	1.010525
4	4	39	12	1225	39.2294	11.117	0.912154
5	6	52	22	796	53.9196	15.2790	6.989716
6	7	19	2	850	18.1935	5.1547	3.256169
7	8	28	0	728	26.1117	7.3895	7.627032
8	11	70	23	542	70.9834	20.168	2.997899
9	12	47	10	1092	46.1347	13.0729	3.192411
10	13	34	16	629	35.7463	10.1321	6.122152

Load Index	IEEE 118-Bus system bus number	Real load of IEEE 118 bus system (MW)	Reactive load of IEEE 118 bus system (MVAR)	Texas system bus number	Mean Real load of Texas system (MW)	Mean Reactive load of Texas system (MVAR)	Orthogonal Difference (MW)
11	14	14	1	156	13.2778	3.7597	2.852647
12	15	90	30	1088	91.1413	25.8227	4.330349
13	16	25	10	762	25.8251	7.3194	2.804699
14	17	11	3	987	11.0073	3.1107	0.111019
15	18	60	34	381	63.4198	18.0132	16.34846
16	19	45	25	464	48.8899	13.9379	11.72609
17	20	18	3	590	17.3915	4.9252	2.019082
18	21	14	8	294	14.9588	4.2394	3.880843
19	22	10	5	905	10.789	3.0742	2.081081
20	23	7	3	1292	7.22466	2.0472	0.978904
21	24	13	0	98	12.0059	3.3984	3.54081
22	27	71	13	908	69.4256	19.6671	6.850562
23	28	17	7	1321	17.5553	4.9749	2.099836
24	29	24	4	407	23.3147	6.6042	2.692891
25	31	43	27	978	46.7610	13.2561	14.24913
26	32	59	23	1158	60.5819	17.1681	6.042598

Load Index	IEEE 118-Bus system bus number	Real load of IEEE 118 bus system (MW)	Reactive load of IEEE 118 bus system (MVAR)	Texas system bus number	Mean Real load of Texas system (MW)	Mean Reactive load of Texas system (MVAR)	Orthogonal Difference (MW)
27	33	23	9	328	23.5907	6.6854	2.388738
28	34	59	26	1019	61.5085	17.427	8.932256
29	35	33	9	825	32.9767	9.2945	0.295503
30	36	31	17	1209	33.0487	9.3667	7.903436
31	39	27	11	1083	27.8511	7.8936	3.220887
32	40	66	23	717	66.8014	18.931	4.147033
33	41	37	10	706	36.9012	10.4499	0.460708
34	42	96	23	1077	95.5490	27.0730	4.097985
35	43	18	7	858	18.5881	5.2691	1.828009
36	44	16	8	829	16.8980	4.7962	3.327236
37	45	53	22	1220	54.8055	15.5306	6.716604
38	46	28	10	207	28.4210	8.0526	1.992299
39	47	34	0	141	31.6466	8.9645	9.268361
40	48	20	11	199	21.4250	6.0735	5.128423
41	49	87	30	317	87.9708	24.9284	5.16364
42	50	17	4	341	16.8321	4.7686	0.786752

Load Index	IEEE 118-Bus system bus number	Real load of IEEE 118 bus system (MW)	Reactive load of IEEE 118 bus system (MVAR)	Texas system bus number	Mean Real load of Texas system (MW)	Mean Reactive load of Texas system (MVAR)	Orthogonal Difference (MW)
43	51	17	8	324	17.8279	5.0536	3.060445
44	52	18	5	852	18.0200	5.1094	0.110043
45	53	23	11	71	24.05745	6.8189	4.31274
46	54	113	32	444	113.0483	32.0317	0.057844
47	55	63	22	667	64.241	18.1983	3.999103
48	56	84	18	1002	82.2678	23.3067	5.582281
49	57	12	3	1248	11.9240	3.37663	0.384219
50	58	12	3	1248	11.9240	3.37662	0.384219
51	59	277	113	1113	174.2157	49.3634	120.8893
52	60	78	3	1005	72.9071	20.6542	18.37416
53	62	77	14	569	74.9292	21.2244	7.51535
54	66	39	18	502	40.7289	11.5539	6.673905
55	67	28	7	840	27.7156	7.8521	0.898304
56	70	66	20	301	66.2442	18.7669	1.257022
57	72	12	0	830	11.0218	3.1127	3.262867
58	73	6	0	924	5.4557	1.5344	1.628094

Load Index	IEEE 118-Bus system bus number	Real load of IEEE 118 bus system (MW)	Reactive load of IEEE 118 bus system (MVAR)	Texas system bus number	Mean Real load of Texas system (MW)	Mean Reactive load of Texas system (MVAR)	Orthogonal Difference (MW)
59	74	68	27	1028	70.0792	19.8583	7.43813
60	75	47	11	628	46.3809	13.1409	2.22864
61	76	68	36	792	72.6737	20.6375	16.05765
62	77	61	28	381	63.4198	18.0132	10.27576
63	78	71	26	792	72.6737	20.6375	5.617564
64	79	39	32	909	44.2945	12.5607	20.14735
65	80	130	26	1043	127.237	36.0498	10.42268
66	82	54	27	771	56.3149	16.0245	11.2169
67	83	20	10	1208	21.0805	5.9737	4.168672
68	84	11	7	702	12.0405	3.4119	3.735922
69	85	24	15	885	26.4375	7.4996	7.886471
70	86	21	10	564	22.1497	6.2774	3.896028
71	88	48	10	892	47.2730	13.3758	3.453223
72	90	163	42	283	162.6423	46.0853	4.10098
73	91	10	0	881	9.5266	2.65173	2.693659
74	92	65	10	1129	62.5012	17.7083	8.103255

Load Index	IEEE 118-Bus system bus number	Real load of IEEE 118 bus system (MW)	Reactive load of IEEE 118 bus system (MVAR)	Texas system bus number	Mean Real load of Texas system (MW)	Mean Reactive load of Texas system (MVAR)	Orthogonal Difference (MW)
75	93	12	7	887	12.9437	3.6720	3.459157
76	94	30	16	1184	31.898	9.0398	7.214255
77	95	42	31	978	46.7610	13.2562	18.37145
78	96	38	15	1225	39.2294	11.1171	4.072819
79	97	15	9	984	16.2099	4.6056	4.557849
80	98	34	8	285	33.5045	9.4930	1.573082
81	99	42	0	478	38.8108	10.9926	11.44594
82	100	37	18	623	38.8607	11.0107	7.232703
83	101	22	15	608	24.6084	6.9824	8.431155
84	102	5	3	601	5.4522	1.5495	1.519284
85	103	23	16	349	25.6015	7.2550	9.123684
86	104	38	25	531	41.9577	11.917	13.66853
87	105	31	26	465	34.5231	9.8637	16.51639
88	106	43	16	688	43.9733	12.4581	3.673194
89	107	50	12	433	49.5594	13.9735	2.022099
90	108	2	1	48	2.09208	0.5946	0.415661

Load Index	IEEE 118-Bus system bus number	Real load of IEEE 118 bus system (MW)	Reactive load of IEEE 118 bus system (MVAR)	Texas system bus number	Mean Real load of Texas system (MW)	Mean Reactive load of Texas system (MVAR)	Orthogonal Difference (MW)
91	109	8	3	1324	8.2236	2.3315	0.704903
92	110	39	30	909	44.2945	12.5607	18.22523
93	112	68	13	301	66.2442	18.7669	6.028287
94	113	6	0	924	5.4557	1.5344	1.628094
95	114	8	3	1324	8.2236	2.3315	0.704903
96	115	22	7	564	22.1497	6.2774	0.737868
97	116	184	0	1016	170.285	48.2441	50.15552
98	117	20	8	741	20.6832	5.8643	2.24232
99	118	33	15	465	34.5231	9.8637	5.357329

Therefore, the distribution of the load pattern for the real and reactive loads for the IEEE 118-bus system can be obtained by matching the loads of the IEEE 118-bus system with the synthetic 2000-bus Texas system and shifting the distribution of the loads of the synthetic Texas system by the respective differences in means.

The load data points are now sampled from the obtained distribution of IEEE 118-bus system loads. These sampled load data points are fed into MATPOWER to obtain the

ACOPF solution for different discrete load values. To perform state estimation for a real system, the measurements for training and validation are generated using historical load data and the data for testing can be the real time PMU measurements. However, since the real time measurements for the IEEE 118-bus system are not available, the load data samples required for the training, validation and testing will all be obtained from the same load distribution.

Due to shifting of the load distribution from the loads of synthetic Texas system, the sampled points of the load of the IEEE 118-bus system might be negative. The load data set with negative load values (predominantly reactive load) are neglected for the data generation process. These sampled and filtered load data points are fed into MATPOWER to obtain the ACOPF solution for different discrete load values. The ACOPF solution gives the true voltage values of the IEEE 118-bus system. The true current values can be generated using the true voltage values and the line parameters of the system.

3.6 Generation of Currents:

The lines in the transmission system can be modelled using a pi-model [84] . A transmission line connecting two nodes with a line series impedance of $R + jX$ and a susceptance value of $j2B$ will be modeled as shown in Fig 3.7.

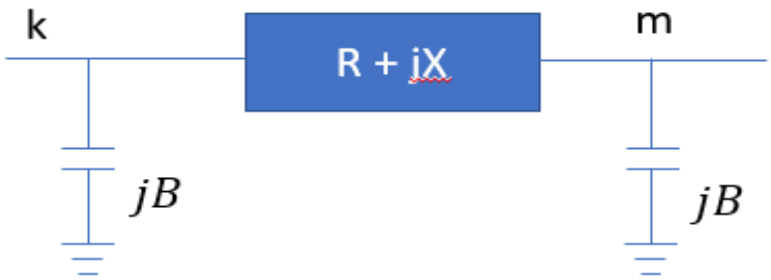

Figure 3.7: A Pi-Model Equivalent Circuit of a Transmission Line

The node k denotes the sending end node and m denotes the receiving end node. The true voltage values at both the nodes are already obtained and the values of R, X and B for IEEE 118-bus system are available in the MATPOWER. A PMU placed at bus k measures the line current leaving out of node k towards the series branch and the shunt component. The line current leaving node k will be sum of the currents in the series branch with impedance $R + jX$ and the current in the shunt branch jB. If V_k and V_m are the true voltages at nodes k and m, respectively, then the current I_{km} leaving the node k towards the branch km is:

$$I_{km} = \frac{(V_k - V_m)}{(R + jX)} + (V_k) * jB \tag{3.16}$$

Similarly, a PMU placed at bus m will measure the difference of the current entering the node m from the series branch km and the corresponding shunt branch. If I_{mk} denotes the current entering node m from node k,

$$I_{mk} = \frac{(V_k - V_m)}{(R + jX)} - (V_m) * jB \tag{3.17}$$

Using the above two formulas, the current entering the node towards all the branches or leaving the node using all the branches can be calculated.

Two buses in an IEEE 118-bus system can also be connected by a transformer. The pi model equivalent circuit for two buses connected by a transformer having turns ratio a is given by:

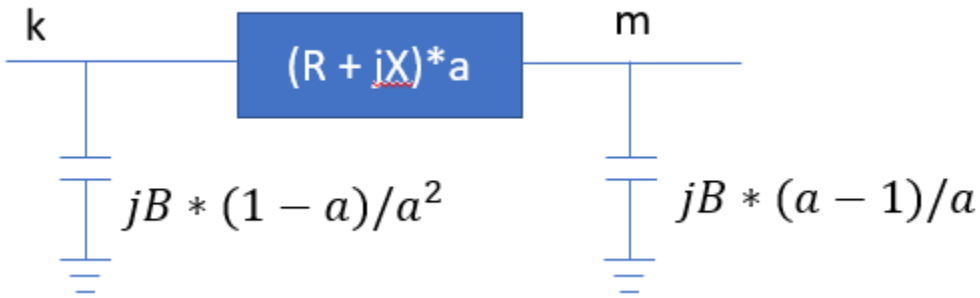

Figure 3.8: A Pi-Model Equivalent Circuit of a Transformer

The currents flowing into and out of the transformers can be calculated by modifying 3.16 and 3.17, appropriately.

Therefore, the true values of voltages and currents of the entire network can be obtained using the available hourly load data. The neural network formulation in this thesis is performed using the training, validation and testing samples obtained from the same distribution. The number of samples obtained for the different types of data after solving the ACOPF are as follows:

Table 3.4: Number of Data Samples for Different Datasets

Type of Data	**Number of Samples**
Training	8500
Validation	2700
Testing	1000

Therefore, the data generation for the neural network consists of generating many samples of true values of voltage magnitude and angle of all buses and true values of currents entering or leaving all the branches in the network. The errors can then be added to these true values to obtain the measurements for the neural network.

The PMU device is generally very accurate and are required to be calibrated to produce an error of less than 1% Total Vector Error (TVE) under steady-state conditions. The TVE defined in the IEEE C37.118.1 [85] standard is mathematically defined as:

$$TVE = \sqrt{\frac{\left(V_r - \widehat{V_r}\right)^2 + \left(V_i - \widehat{V_i}\right)^2}{\widehat{V_r}^2 + \widehat{V_i}^2}} \tag{3.18}$$

The operation of the neural network-based state estimator is demonstrated initially with the measurements obtained by adding error values chosen randomly from a Gaussian distribution. The parameters of the Gaussian distribution chosen for the error are mentioned in Table 3.5 below.

Table 3.5: Gaussian Error Measurement Parameters

Measurement type	Error Mean	Error Standard Deviation
Magnitude (voltages and currents)	0	1/3 %
Angle (voltages and currents)	0	0.573°/3

Fig 3.9 shows the PDF of a Gaussian distribution. If the Gaussian distribution is plotted, it can be observed that a random value chosen from the distribution has a probability of 99.7% to lie in the range $\pm 3\sigma$.

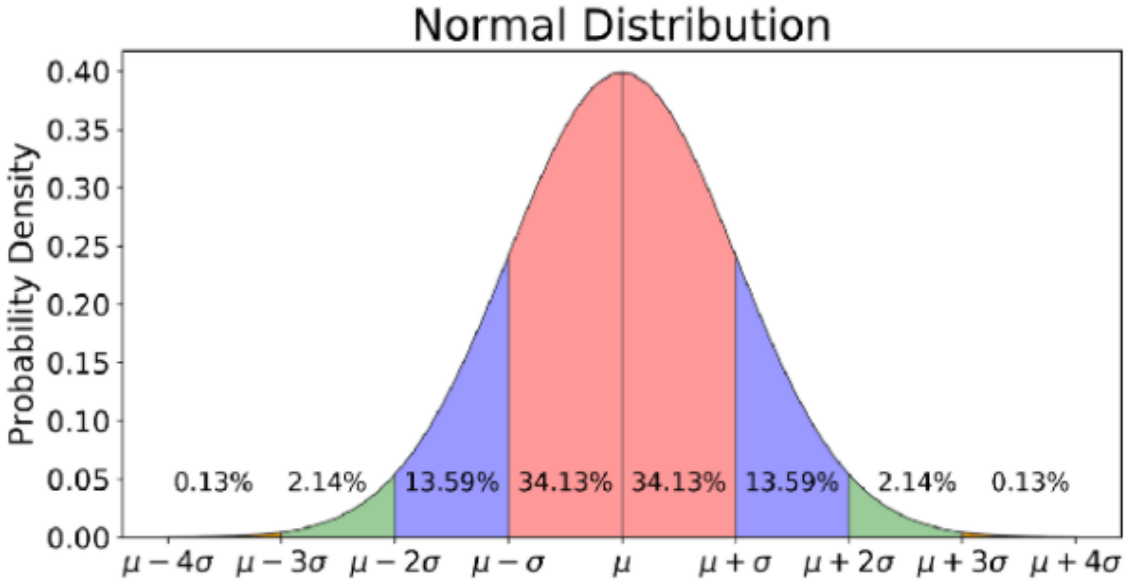

Figure 3.9: PDF of a Gaussian Distribution

Thus, the standard deviation of 1/3% corresponds to the maximum 3σ error in magnitude measurement when the angle measurement is error free for a 1% TVE. Similarly, the standard deviation of $0.573°/3$ corresponds to the maximum 3σ error in angle measurement

when the magnitude measurement is error free for a 1% TVE. Additionally, the TVE is calculated after choosing the random error values from the Gaussian distribution. If the TVE exceeds the 1% criteria, these error values are discarded, and new error values are randomly chosen from the distribution. This process is repeated until the TVE condition is satisfied.

3.7: Neural Network Overview:

The equation relating an error induced measurement ($z = Hx + e$) and the state (x) of a system is given by:

$$z = Wx + b \tag{3.19}$$

The neural network can identify the weights W and bias b mapping the measurements and the states using various samples of features, label data pair obtained from the distributions of hourly load data in the training and the validation phase. The states for any new set of measurements in the testing data can be predicted using the estimated W and b. Fig 3.10 below demonstrates a simple representation of a multi layered neural network (MLNN).

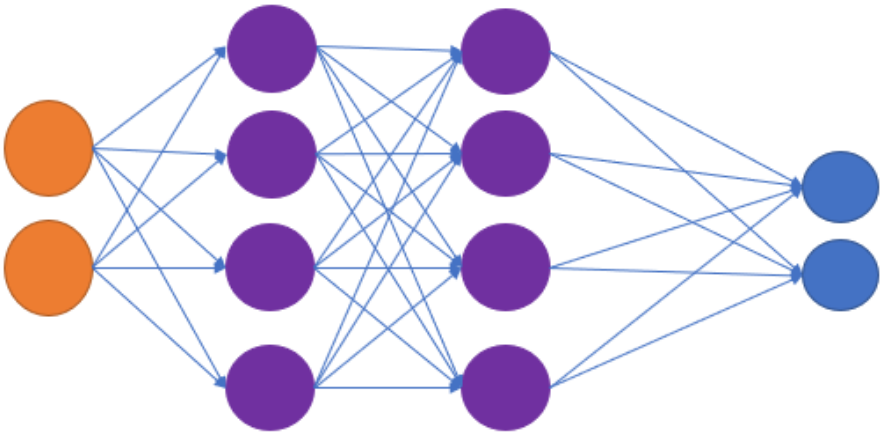

Figure 3.10: Demonstration of a Multi Layered Neural Network

Consider a neural network with one output and no hidden layers as shown in the figure below.

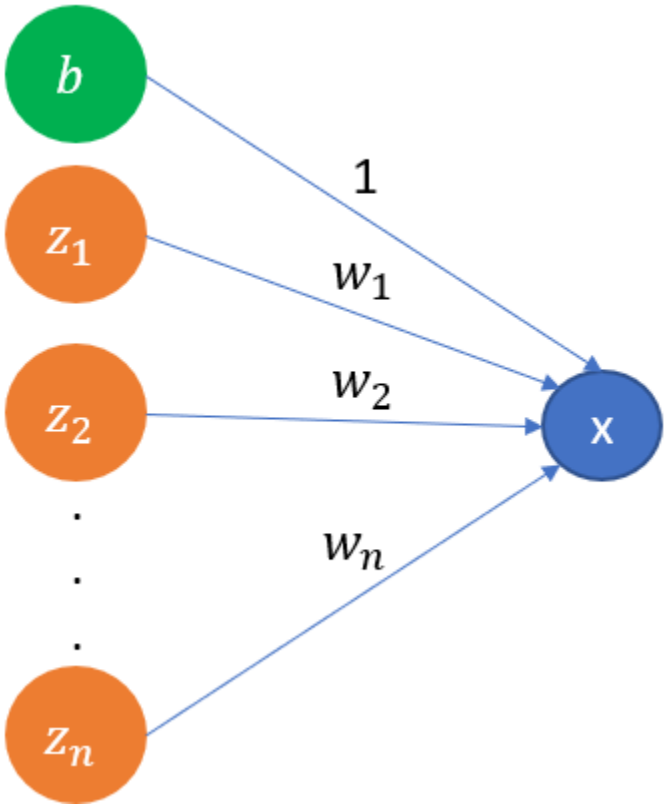

Figure 3.11: Neural Network with No Hidden Layers

Let $Z = \{z_1, z_2, \ldots . z_n\}$ correspond to the features of the neural network and $W = \{w_1, w_2, \ldots . w_n\}$ represent the weights of the edges connecting the input to the output. The b corresponds to the bias, which can also be modeled as one of the neural network inputs connected to the next layer with weight 1.

The output x is given by:

$$x = g(w_1 z_1 + w_2 z_2 + w_3 z_3 + \cdots + w_n z_n + b) \tag{3.20}$$

The output x is the weighted sum of inputs of the neural network applied over an activation function, which is mostly non-linear. Some of the predominantly used activation functions include Sigmoid, tanh, ReLU, etc. This process of computing the values of nodes in a layer with the values of nodes in the previous layer is called forward propagation.

The same concept is applicable in case of neural network with multiple hidden layers too. Consider a neural network with 3 input nodes, 4 nodes in the hidden layer and 2 outputs as shown in the Fig 3.12 below:

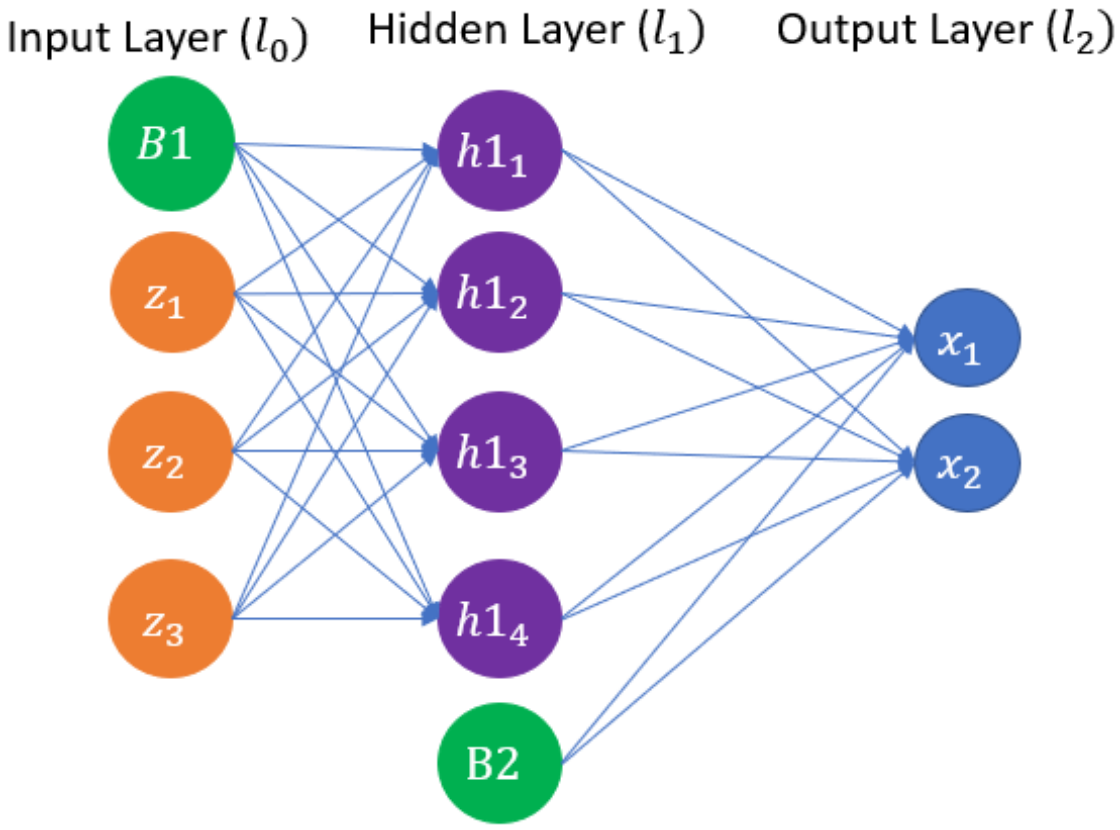

Figure 3.12: Neural Network with One Hidden Layer

Let $W1 = \begin{bmatrix} w1_{11} & w1_{12} & w1_{13} \\ w1_{21} & w1_{22} & w1_{23} \\ w1_{31} & w1_{32} & w1_{33} \\ w1_{41} & w1_{42} & w1_{43} \end{bmatrix}$ denote the elements of the weight matrix connecting the input layer (l_0) with features $Z = [z_1\ z_2\ z_3]$ and the hidden layer (l_1) with nodes $h1 = [h1_1\ h1_2\ h1_3\ h1_4]$. Let $W2 = \begin{bmatrix} w2_{11} & w2_{12} & w2_{13} & w2_{14} \\ w2_{21} & w2_{22} & w2_{23} & w2_{24} \end{bmatrix}$ denote the elements of the weight matrix connecting the hidden layer (l_1) and the output layer (l_2) with nodes $X = [x_1 \quad x_2]$. The biases $B1 = [b1_1 \quad b1_2 \quad b1_3 \quad b1_4]$ and $B2 = [b2_1 \quad b2_2 \quad b2_3 \quad b2_4]$ can be assumed to be connected to all the nodes of the next layer with weights value of 1.

The equations for forward propagation of the neural network is given by:

$$h1_1 = g_1(z_1 w1_{11} + z_2 w1_{12} + z_3 w1_{13} + b1_{11}) \tag{3.21}$$

$$h1_2 = g_1(z_1 w1_{21} + z_2 w1_{22} + z_3 w1_{23} + b1_{21}) \tag{3.22}$$

$$h1_3 = g_1(z_1 w1_{31} + z_2 w1_{32} + z_3 w1_{33} + b1_{31}) \tag{3.23}$$

$$h1_4 = g_1(z_1 w1_{41} + z_2 w1_{42} + z_3 w1_{43} + b1_{41}) \tag{3.24}$$

Once the values of the hidden layers are calculated, the values in the output layer is given by:

$$x_1 = g_2(h1_1 w2_{11} + h1_2 w2_{12} + h1_3 w2_{13} + h1_4 w2_{14} + b2_{11}) = g_2(a2_1) \tag{3.25}$$

$$x_2 = g_2(h1_1 w2_{21} + h1_2 w2_{22} + h1_3 w2_{23} + h1_4 w2_{24} + b2_{21}) = g_2(a2_2) \tag{3.26}$$

where g_1 denotes the activation function of the nodes in the hidden layer and g_2 denotes the activation function of the nodes in the output layer.

The values of input matrix are known values. Initially, the weights and bias vectors are assigned with a random value. The value of the nodes in the hidden layers and output layer will be computed.

The computed output will be very different from the labeled output with the weights being assigned a random value. However, the output predicted by the neural network must become as close as possible to the labeled output for an accurate estimation. This prediction error of the neural network can be modeled using a loss function. Some of the commonly used loss functions for the regression problems include mean absolute error (MAE), mean square error (RMSE), mean absolute percentage error (MAPE), etc. For instance, the mean square error loss function for the neural network shown in Fig 3.12 is given by:

$$L(x) = \frac{1}{2} \left|(x_1 - x_{1p})^2 + (x_2 - x_{2p})^2\right| \tag{3.27}$$

This loss function will be a function of weight and bias of the neural network. This loss function can be minimized using an iterative gradient descent algorithm (or other sophisticated algorithms).

$$W1 := W1^0 - \alpha \frac{dL}{dW1} \tag{3.28}$$

$$W2 := W2^0 - \alpha \frac{dL}{dW2} \tag{3.29}$$

$$B1 := B1^0 - \beta \frac{dL}{dB1} \tag{3.30}$$

$$B2 := B2^0 - \beta \frac{dL}{dB2} \tag{3.31}$$

The variables $W1^0, W2^0, B1^0, B2^0$ denote the weight matrices and bias vectors during the previous iteration and W1, W2, B1 and B2 denotes the computed weights and biases in the current iteration after backpropogation. α and β denote the learning rates which will be decided by the user.

The derivative of the loss function with respect to the weights can be calculated in the following manner:

$$\frac{dL}{da2_1} = \frac{dL}{dx_1} * \frac{dx_1}{da2_1} = \frac{dL}{dx_1} * g_2' \tag{3.32}$$

$$\frac{dL}{da2_2} = \frac{dL}{dx_2} * \frac{dx_2}{da2_2} = \frac{dL}{dx_2} * g_2' \tag{3.33}$$

The derivatives with respect to the loss function will be a term dependent on the loss function.

$$\left[\frac{dL}{dW2}\right] = \begin{bmatrix} \frac{dL}{da2_1} \\ \frac{dL}{da2_2} \end{bmatrix} * [h1_1 \quad h1_2 \quad h1_3 \quad h1_4] = \frac{dL}{da2} * h1^T \tag{3.34}$$

$$\left[\frac{dL}{dB2}\right] = \begin{bmatrix} \frac{dL}{da2_1} \\ \frac{dL}{da2_2} \end{bmatrix} = \frac{dL}{da2} \tag{3.35}$$

Similarly, the slope of the weights and bias vectors connecting the input and the hidden layer are:

$$\left[\frac{dL}{dW1}\right] = \begin{bmatrix} \frac{dL}{da1_1} \\ \frac{dL}{da1_2} \\ \frac{dL}{da1_3} \\ \frac{dL}{da1_4} \end{bmatrix} * \begin{bmatrix} z_1 & z_2 & z_3 \end{bmatrix} = \frac{dL}{da1} * Z^T \tag{3.36}$$

$$\left[\frac{dL}{dB1}\right] = \begin{bmatrix} \frac{dL}{da1_1} \\ \frac{dL}{da1_2} \\ \frac{dL}{da1_3} \end{bmatrix} = \frac{dL}{da1} \tag{3.37}$$

This can be extended to a MLNN with l hidden layers. The gradient of the weights W_l and bias vectors B_l connecting any two layers l and $l+1$ is given by:

$$\left[\frac{d(L)}{d(W_l)}\right] = = \frac{d(L)}{d(a_l)} * (h_{l-1})^T \tag{3.38}$$

$$\left[\frac{d(L)}{d(B_l)}\right] = = \frac{d(L)}{d(a_l)} \tag{3.39}$$

where a_l and h_{l-1} are vectors in layer l and $l-1$ related in the following manner:

$$a_l = W_l * h_{l-1} + B_l \tag{3.40}$$

Therefore, an initial output of the neural network is computed using forward propagation using randomly chosen values of weights and biases. The error in the neural network prediction is modeled using a loss function. The neural network predicts the best results when the computed value of the loss function is minimal. The error computed by the loss function is then backpropagated to readjust the weights. This process of forward and back

propagation is continued iteratively until a certain number of iterations are reached or the absolute change in the values computed by the loss function is lesser than a certain threshold. The set of weight matrices and bias vectors obtained at the end of this process computes the output closest to the labeled output. Therefore, this set of weight matrices and bias vectors obtained at the end of this process represents the mapping between the features (measurements) and the labels (states) of the network.

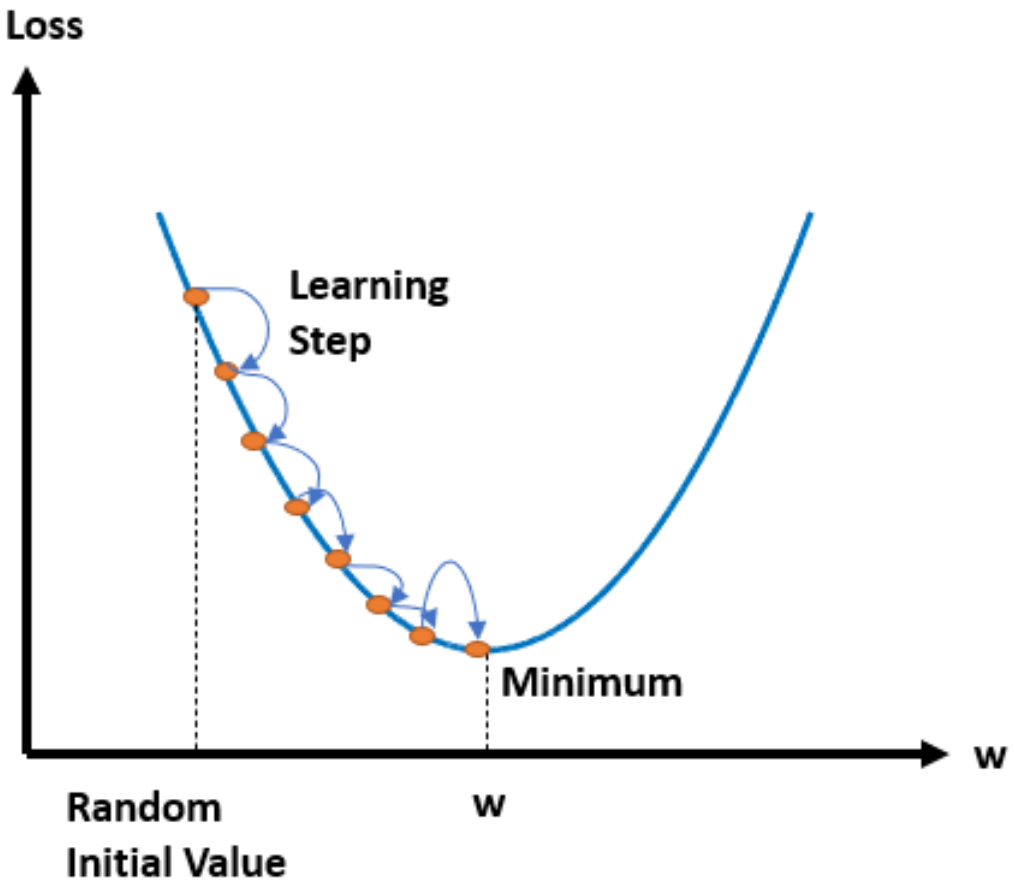

Figure 3.13: Graph of Loss Function Versus Weight

The graph in Fig 3.13 above is a generic representation of the loss function as a function of a weight of a neural network. A lower learning rate α chosen for the neural network, will take more time for weights to converge. A higher learning rate α chosen for the neural

network will take less time for weights to converge, however it runs into the problem of the loss function not reaching a global minimum, and possibly resulting even in divergence. Thus, there is a trade-off between the time taken for training and the accuracy of training, while choosing the value of the learning rate. The value chosen for the learning rate controls the training process and hence it is termed as a hyperparameter. Choosing the value of hyperparameters is an extremely crucial task in training and testing the neural network.

3.8: Neural Network Implementation for the IEEE 118-Bus System:

The neural network is capable of identifying the joint probability distribution between any set of measurements and the corresponding true values of the system. This joint probability distribution represents the relationship between the measurements and the states of the system. The feature, label pair data sets for training, validation and testing required for the neural network is already obtained using the procedure discussed earlier. The next step is identifying the architecture of the neural network to be used for the application.

Fig 3.14 below shows the architecture of the neural network that can be used for identifying the mapping determining the relationship between the measurements and the states. All the buses and branches are assumed to be completely observed by PMUs for the current set-up resulting in a system that has redundant observability. The working of the neural network-based state estimation in case of a partial observability will be discussed afterwards. The input measurements corresponding to the features of this neural network are the complex voltage measurements of all buses and the complex current measurements of all the branches. The topology of the IEEE 118-bus system is obtained from

MATPOWER. The IEEE 118-bus system has 118 buses and 186 branches (including the connections between a bus and a transformer).

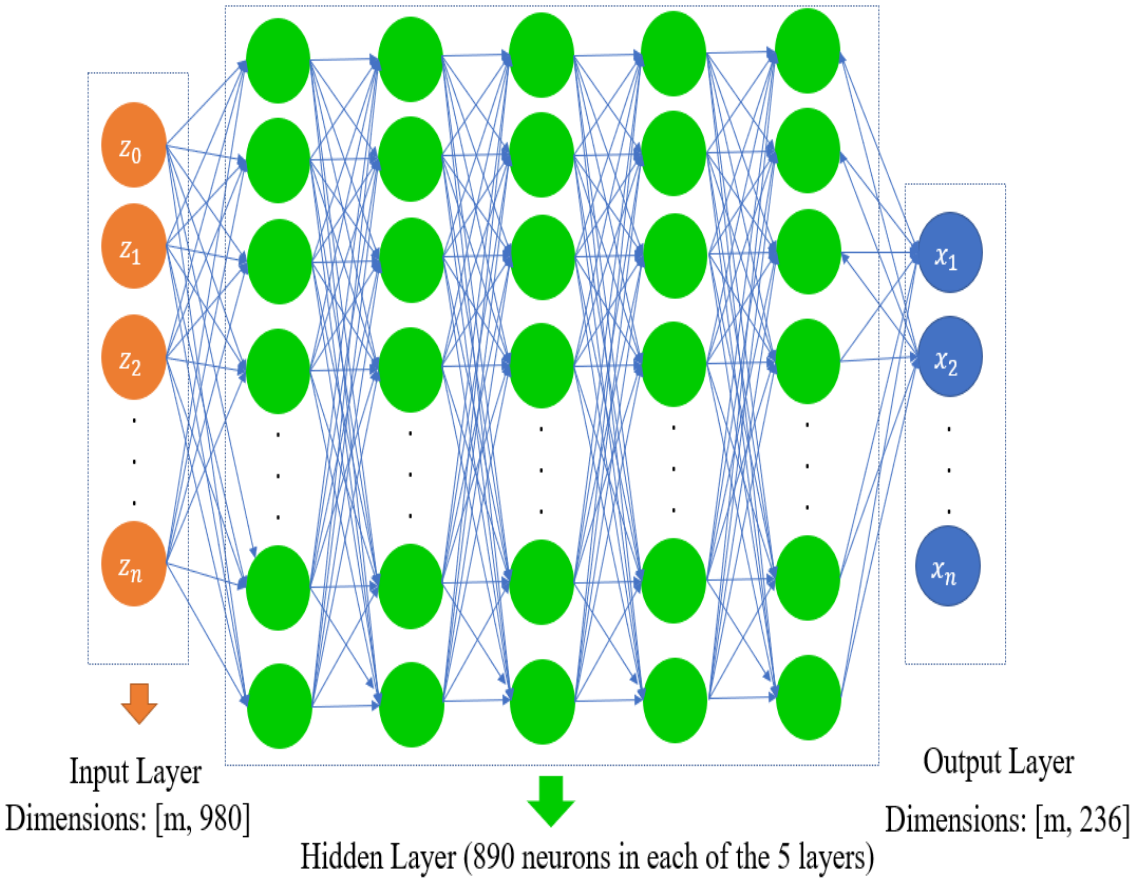

Figure 3.14: Architecture of the Proposed Neural Network for State Estimation

$$Number\ of\ measurements$$
$$= Number\ of\ complex\ voltage\ measurements$$
$$+ Number\ of\ complex\ current\ measurements\ entering\ the\ branch$$
$$+ Number\ of\ complex\ current\ measurements\ leaving\ the\ branch$$
$$= (118 * 2) + (186 * 2) + (186 * 2) = 980$$

A complex measurement corresponds to a magnitude and an angle measurement. The size of the input data will be $[m, 980]$ where m corresponds to the number of samples of input.

The value of m will be different for the training, validation, and testing datasets. Therefore, each row of the data corresponds to different samples of a feature and each column corresponds to different input features for the neural network.

The next step is to choose the number of hidden layers and the size of each layer. The number of hidden layers and size of each layer are also hyperparameters which controls the training of data by the neural network.

Hidden Layer Dimensions: A deep neural network with 5 hidden layers are chosen. There are no particular best values to choose for the number of nodes in each hidden layer. However, the number of nodes preferred for the hidden layer are roughly two-thirds the size of the input layer plus the size of the output layer [86] . The number of features (input) and labels (output) respectively are 980 and 236. Therefore, 5 hidden layers with 890 nodes in each hidden layer are chosen for the neural network architecture.

Activation function: The next hyperparameter to be decided is the activation function to be used for each hidden layer. Some of the commonly used activation functions include: sigmoid, tanh, ReLU, etc. The sigmoid and tanh activation functions are primarily used for neural network classification problems as they return a value in the range $[-1\ 1]$. The commonly used activation function for neural network regression problems is the Rectified Linear Unit (ReLU) function. The ReLU activation function for an input x is given by:

$$g(x) = \begin{cases} x, & when\ x \geq 0 \\ 0, & when\ x < 0 \end{cases} \tag{3.41}$$

The plot of the ReLU function is shown in Fig 3.15 below. The value of output will be same as input if the sum of the weighted inputs at the node turns out to be positive. It will be zero if the sum of weighted inputs turns out to be negative.

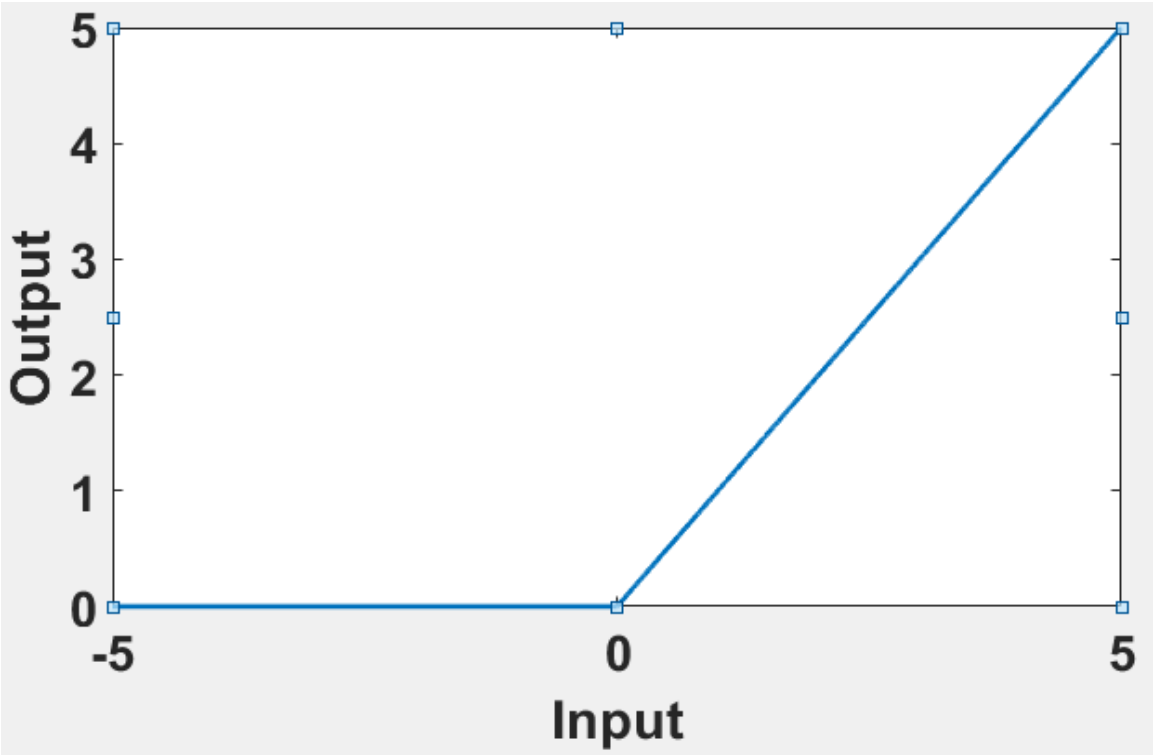

Figure 3.15: Demonstration of ReLU Activation Function

The ReLU function is used as the activation function for the all the connections between the input layer and the first hidden layer and also between any two hidden layers. A linear function is used as the activation function for the connections between the final hidden layer and the output layer. The linear activation function for an input x is given by:

$$g(x) = x$$

The plot of a linear activation function is shown in the Fig 3.16 given below:

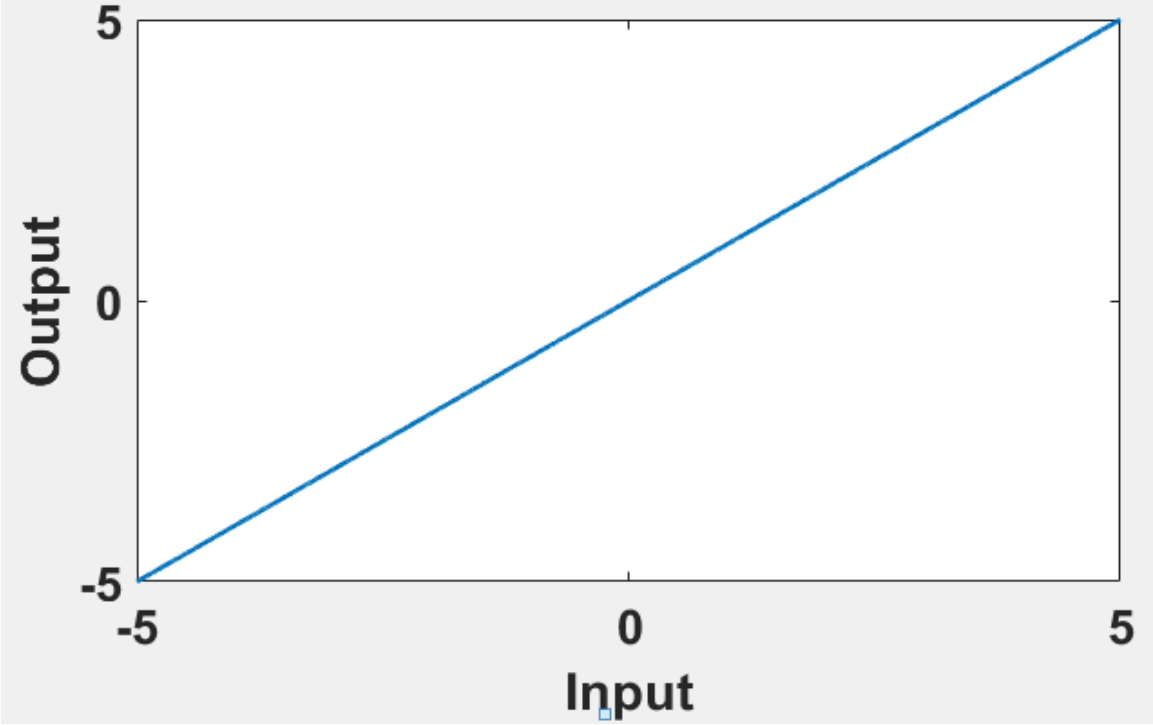

Figure 3.16: Demonstration of Linear Activation Function

Loss function: The loss function that will be used for this application is also the commonly used loss function for the neural network regression applications. The difference between the predicted output and the labelled output at the end of each iteration is modeled using a mean squared error (MSE) loss function. For a neural network with m samples predicting n outputs of the output layer, the mean squared error is given by:

$$L(x) = \frac{1}{mn} \sum_{i=1}^{m} \left(\left(x_{i1}^{p} - x_{i1} \right)^2 + \left(x_{i2}^{p} - x_{i2} \right)^2 + \cdots + \left(x_{in}^{p} - x_{in} \right)^2 \right) \qquad (3.42)$$

The values x_{ij}^{p} and x_{ij} denotes the predicted value of the i^{th} sample, j^{th} output and the labeled true value, respectively. The training and validation loss will be calculated at the end of each iteration.

Optimization algorithm: The adjustment of weights and bias vectors in each iteration using a gradient descent algorithm was demonstrated earlier. The formula used for calculating the weights W and biases B using an iterative gradient descent algorithm is given by:

$$W^t := W^{t-1} - \alpha \frac{dL}{dW} \tag{3.43}$$

$$B^t := B^{t-1} - \beta \frac{dL}{dB} \tag{3.44}$$

The optimization algorithm used for this application is Adam, which is a modification of the gradient descent algorithm. The Adam optimization tries to modify the learning rate adaptively to increase the computational speed of the neural network [87] .

Number of Epochs: The number of epochs refer to the number of times the weights of the neural network are updated during the training and validation process. The formulated neural network was run for 120 epochs. The training and the validation loss value, given by the loss function are calculated for each epoch and both the losses are plotted as a function of epoch number.

Early Stopping: An early stopping is introduced into the neural network training process which allows the stopping of the neural network training if the validation loss does not keep on decreasing beyond a specified patience. The neural network is monitored after each epoch and a model of the neural network is saved whenever the validation loss reaches its lowest value. The saved neural network model denotes the weights and bias vectors mapping the features and the labels calculated in the epoch when the validation loss is at its lowest. The patience value refers to the number of epochs up to which the neural

network continues its training without the validation loss going less than the value in the saved model. The neural network was formulated to run for 120 epochs and the number of epochs chosen as the patience was 50. The hyperparameters of the neural network architecture and the values chosen for the different hyperparameters are summarized in the table below:

Table 3.6: Different Hyperparameter Values of the Formulated Neural Network

Hyperparameters	**Values**
Number of hidden layers	5
Number of neurons in the hidden layers	890
Activation Function	ReLU function for all connections between the input and the hidden layer and between 2 hidden layers Linear function for all connections between the final hidden layer and the output layer
Loss Function	Mean Squared Error (MSE)
Optimization algorithm	Adam's Optimization
Learning rate	Adaptive learning rate calculated during the Adam's Optimization
Number of epochs	120
Early Stopping	Monitored Quantity – Validation Loss Patience = 50

A neural network state estimator is formulated with the architecture and the set of hyperparameters described earlier using the inbuilt Python packages, Keras version 2.2.4-tf [88] , Tensorflow version 2.0.0 [89] , and Pandas version 0.25.3 [90] . These packages are integrated using Python version 3.7.7. Four CSV files were fed into the neural network framework for training and validation, while two CSV files were fed into the neural network for testing.

Table 3.7: Dimensions of Different Neural Network Inputs

Input file	**Matrix Dimensions**
Training measurements (NN features)	8500*980
Training states (NN labels)	8500*236
Validation measurements (NN features)	2700*980
Validation states (NN labels)	2700*236
Testing measurements (NN new features)	1000*980
Testing states (NN expected output)	1000*236

The neural network is run for 120 epochs and the training and validation loss obtained from the saved model are:

$$Training\ Loss = \ 5.1122e - 6$$

$$Validation\ Loss = \ 4.3374e - 6$$

Fig 3.17 below represents the plot of the training loss and validation loss obtained after each epoch.

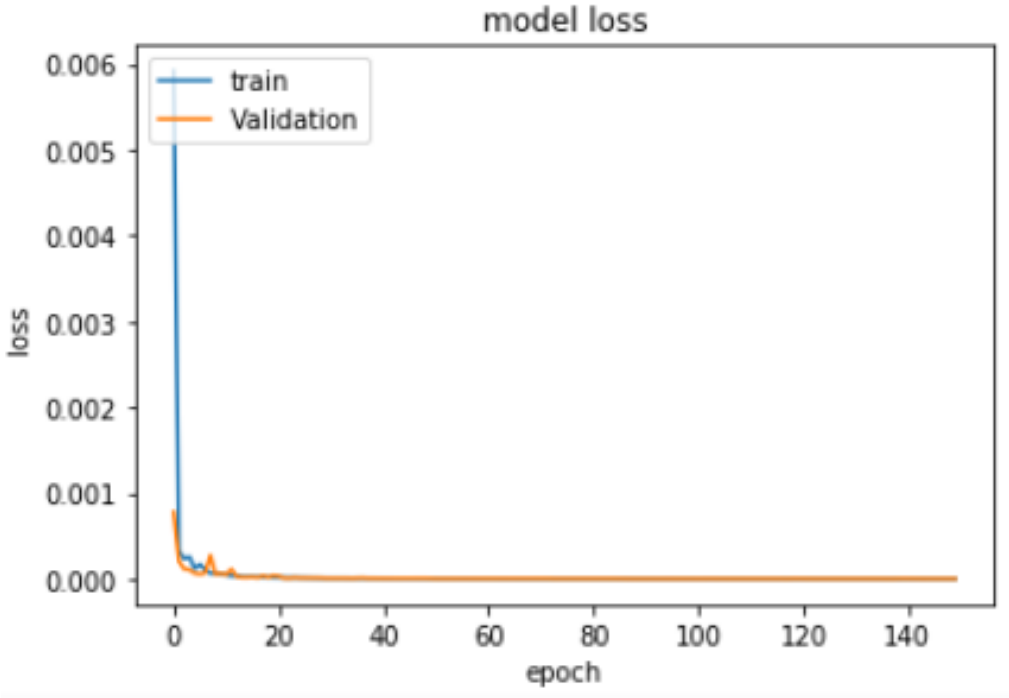

Figure 3.17: Plot of Neural Network Training and Validation Loss

3.9 Error Metrics:

The results of the neural network-based state estimator will be analyzed using the following metrics:

a) Voltage Magnitude by Mean Absolute Error (MAE): The Voltage Magnitude MAE calculates the absolute difference between the predicted and labeled voltage magnitude values and calculates the average of all the calculated absolute differences. If V_{ijm} and V_{ijm}^{p} denotes the labeled and the average estimated values of all samples, respectively, of the i^{th} output, j^{th} sample among n total outputs and k samples, the voltage magnitude MAE is given by:

$$Voltage\ magnitude\ MAE = \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{k}\left|V_{ijm} - V_{ijm}^{p}\right| \qquad (3.45)$$

b) Voltage Angle by Mean Absolute Error (MAE): The Voltage Angle MAE calculates the absolute difference between the predicted and labeled voltage angle values and calculates the average of all the calculated absolute differences. If V_{ia} and V_{ia}^{p} denotes the labeled and estimated values of all samples, respectively, of the i^{th} output, j^{th} sample among n total outputs and k samples, the Voltage Angle MAE is given by:

$$Voltage\ angle\ MAE = \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{k}\left|V_{ija} - V_{ija}^{p}\right| \tag{3.46}$$

c) Voltage by Root Mean Squared Error (RMSE): The Voltage RMSE calculates the square of the difference between the predicted and the labeled states. The states correspond to the complex voltage magnitudes and angles. If V_{ijm}, V_{ija} and V_{ijm}^{p}, V_{ija}^{p} denotes the labeled and estimated magnitude and angle values, respectively, of the i^{th} output, j^{th} sample among n total outputs and k samples, the Voltage RMSE is given by:

$$Voltage\ RMSE = \frac{1}{2mn}\sum_{i=1}^{n}\sum_{j=1}^{k}\left|\left(V_{im} - V_{im}^{p}\right)^{2} + \left(V_{ia} - V_{ia}^{p}\right)^{2}\right| \tag{3.47}$$

The formulated neural network is then tested with the new set of measurements and the predicted state values are obtained. The different error metrics calculated for the above obtained mapping function is shown in the table below.

Table 3.8: Values of Different Metrics of Estimated States with Redundant Observability for Normal Data Set

Voltage magnitude MAE: 9.9609e-04 pu
Voltage angle MAE: 0.1037 degrees
Voltage RMSE: 0.0020

The Voltage magnitude absolute difference and angle absolute difference obtained for different buses averaged across all samples is plotted below:

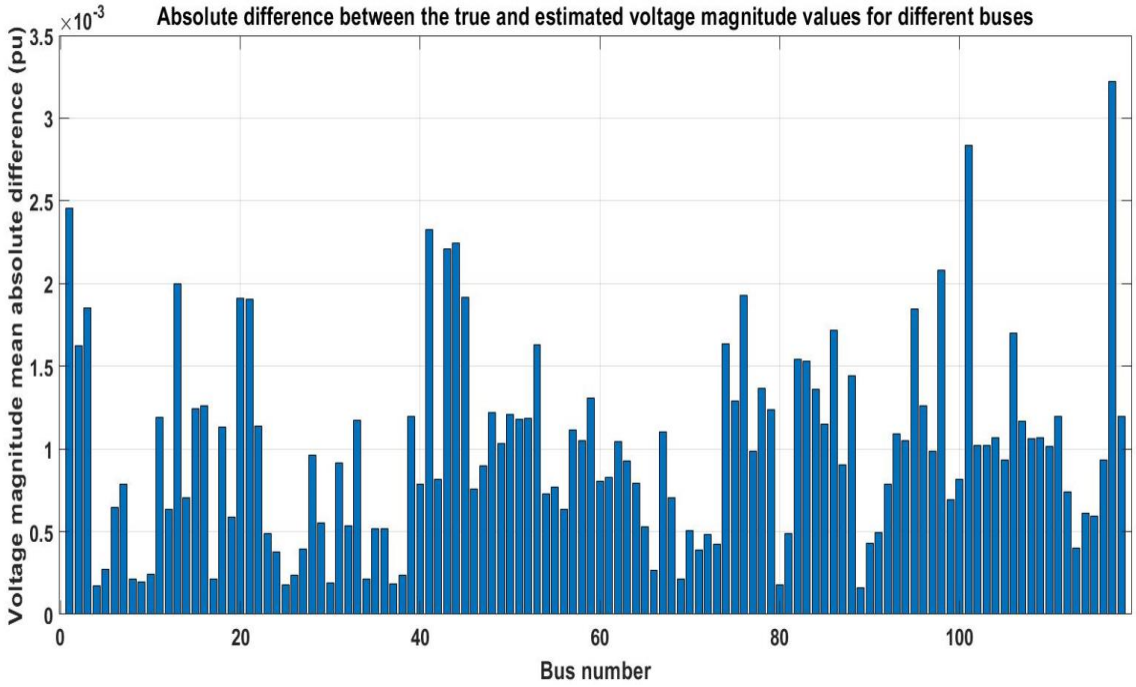

Figure 3.18: Voltage Magnitude Mean Absolute Difference of Estimated and True Values for Different Buses in IEEE 118-Bus System in the Presence of 118 PMUs

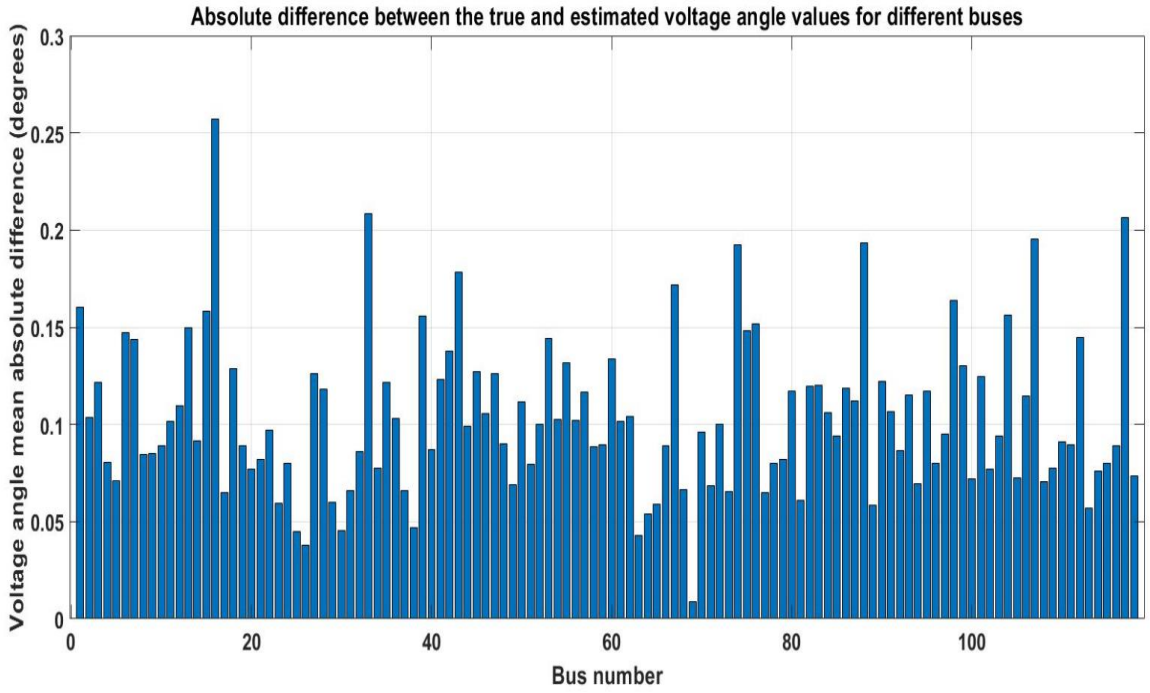

Figure 3.19: Voltage Angle Mean Absolute Difference of Estimated and True Values for Different Buses of IEEE 118-Bus System in the Presence of 118 PMUs

Also, the plot of the estimated complex voltages and the true complex voltages for the first test sample is shown below.

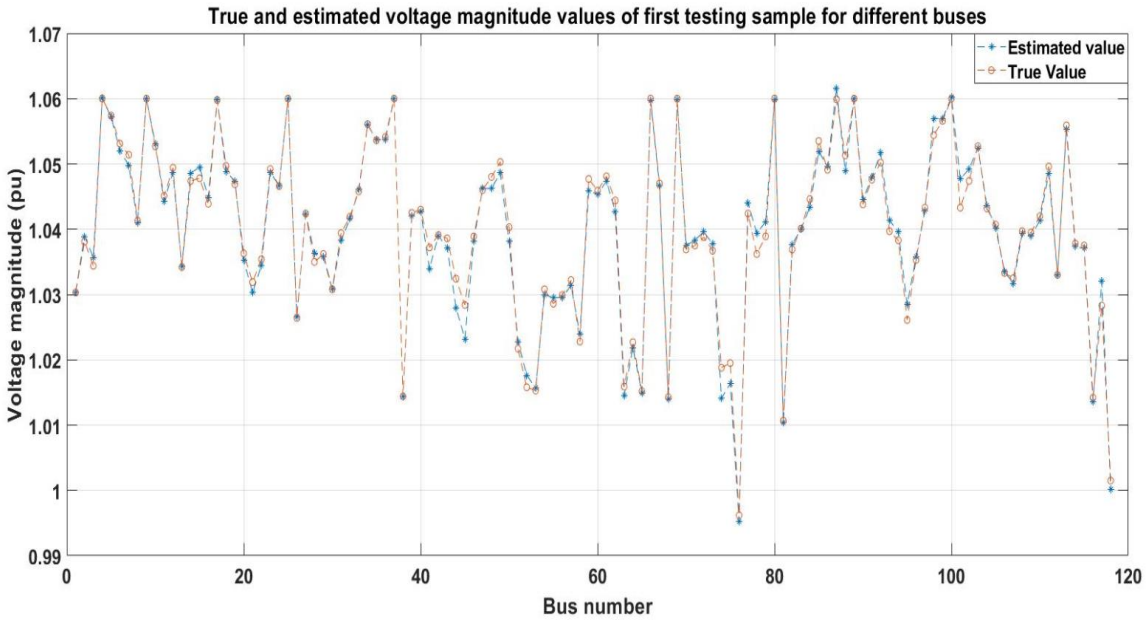

Figure 3.20: Estimated and True Voltage Magnitudes of First Sample for Different Buses of IEEE 118-Bus System in the Presence of 118 PMUs

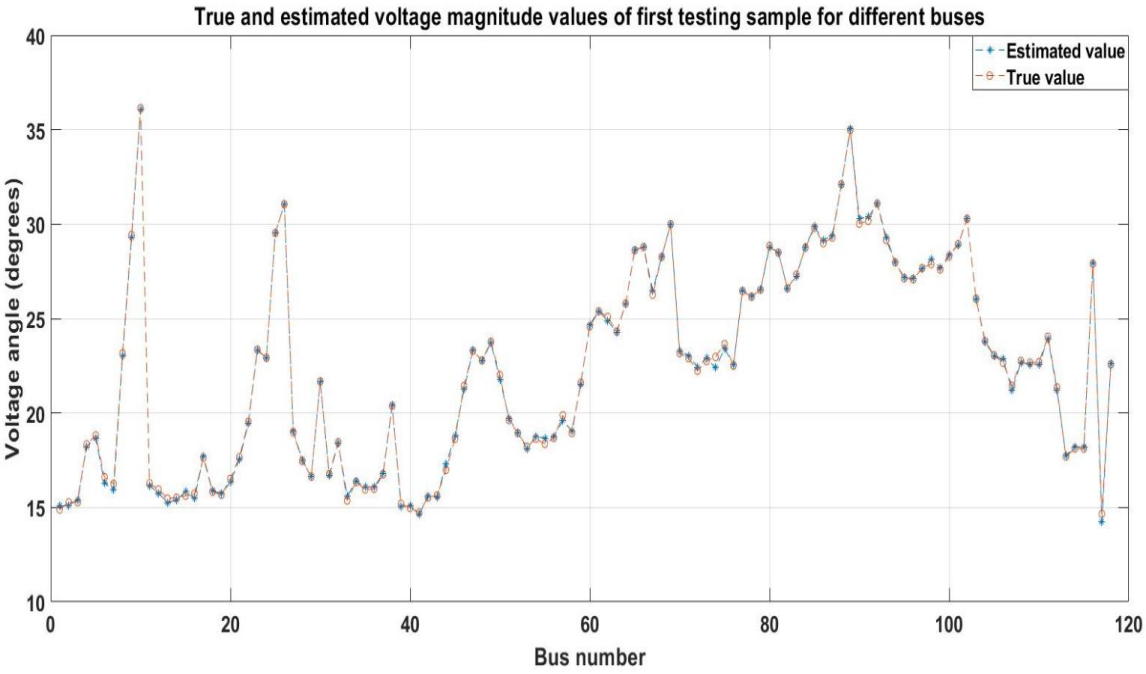

Figure 3.21: Estimated and True Voltage Angles of First Sample for Different Buses of IEEE 118-Bus System in the Presence of 118 PMUs

Next, the neural network state estimator was trained and validated using the same number of input measurements; however, a very large number of training and validation samples were used. The testing was also done for a larger number of samples than the number of samples used for testing earlier. The new dimensions of training, validation, and testing data are given in Table 3.9.

Table 3.9: Dimensions of Large Data Set used as Neural Network Inputs

Input file	**Matrix Dimensions**
Training measurements (NN features)	340000*980
Training states (NN labels)	340000*236
Validation measurements (NN features)	90000*980
Validation states (NN labels)	90000*236
Testing measurements (NN new features)	10000*980
Testing states (NN expected output)	10000*236

The neural network formulation remains unchanged except for the change in the datasets fed into the neural network and the batch size used by the Adam optimizer. The batch size was increased to 64 from 8 previously, to reduce the total time taken for the training and validation of the network. The different error metrics calculated after the testing of the neural network are mentioned in Table 3.10 below.

Table 3.10: Values of Different Metrics of Estimated States with Redundant Observability for Large Data Set

Voltage Magnitude MAE: 8.70 * 10^{-4} pu
Voltage Angle MAE: 0.07908 degrees
Voltage RMSE: 0.0018

It can be observed that the use of a large data set did not result in any significant improvement in the state estimates.

3.10 Linear State Estimation:

The conventional state estimation approach utilizes the topology and line parameter information of the system to estimate the states of the system from the obtained measurements and known models.

The equation relating the erroneous measurements Z with error e and the true voltages X of the system are given by:

$$Z = HX + e \tag{3.48}$$

The matrix H is known as the Jacobian matrix that relates the measurements and the states. The vector Z consists of simulated PMU measurements, the voltages and the currents of the system.

$$Z = \begin{bmatrix} V_{real}^{e} \\ V_{imag}^{e} \\ I_{real}^{e} \\ I_{imag}^{e} \end{bmatrix} = \begin{bmatrix} 1' & 0 \\ 0 & 1' \\ C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} V_{real} \\ V_{imag} \end{bmatrix} + \ e = HX + e \tag{3.49}$$

The values V_{real}^{e}, V_{imag}^{e}, I_{real}^{e}, I_{imag}^{e} corresponds to the PMU voltage measurements in the cartesian form. The values V_{real}, V_{imag} corresponds to the states of the system to be estimated.

The element $1'$ corresponds to the unity matrix element when a PMU is placed at a bus and it corresponds to 0 matrix if a PMU is not placed at a bus. The matrices C_{11}, C_{12}, C_{21}, C_{22} models the relationship between the current measurements and the states of the system and are a function of the line parameters of the system. Therefore, the matrix H is only dependent on the line parameters and topology of the system and is constant for a given topology of the system, making the mapping between the measurements and the states linear. The states of the system $\hat{X}$ can be solved by minimizing the sum of squares of the errors in the measurement.

$$|e|^2 = \frac{\left(Z - H\hat{X}\right)^T \left(Z - H\hat{X}\right)}{W_R} \tag{3.50}$$

In order for $|e|^2$ to be minimum,

$$\frac{d|e|^2}{d\hat{X}} = 0 = \frac{d\left(Z^T W_R^{-1} Z - Z^T W_R^{-1} H\hat{X} - \left(H\hat{X}\right)^T W_R^{-1} Z + H^T H W_R^{-1} \hat{X}^T \hat{X}\right)}{d\hat{X}} \tag{3.51}$$

Solving this equation results in:

$$\hat{X} = (H^T W_R^{-1} H)^{-1} H^T W_R^{-1} Z \tag{3.52}$$

The W_R matrix is given by:

$$W_R = R\, W R^T \tag{3.53}$$

The weight matrix W is a diagonal matrix with diagonal elements equal to the variance (σ^2) of the measurements. The matrix R represents the state rotation matrix used for transforming the weight matrix from polar to the rectangular coordinates. For example, a small change in the measurements of the system from polar ($\Delta V_{polar}, \Delta I_{polar}$) to rectangular coordinates ($\Delta V_{rect}, \Delta I_{rect}$) are obtained in the following manner.

$$\begin{bmatrix} \Delta V_{rect} \\ \Delta I_{rect} \end{bmatrix} = R \begin{bmatrix} \Delta V_{polar} \\ \Delta I_{polar} \end{bmatrix} \tag{3.54}$$

The state rotation matrix R for a system with n buses is given by [14] :

$$R = \begin{bmatrix} \cos\theta_1 & 0 & \cdot & 0 & -|E_1|\sin\theta_1 & 0 & \cdot & 0 \\ 0 & \cos\theta_2 & \cdot & 0 & 0 & -|E_2|\sin\theta_2 & \cdot & 0 \\ \cdot & \cdot & \cdot & 0 & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cos\theta_n & \cdot & \cdot & \cdot & -|E_n|\sin\theta_n \\ \sin\theta_1 & 0 & \cdot & 0 & |E_1|\cos\theta_1 & 0 & \cdot & 0 \\ 0 & \sin\theta_2 & \cdot & 0 & 0 & |E_2|\cos\theta_2 & \cdot & 0 \\ \cdot & \cdot & \cdot & 0 & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \sin\theta_n & \cdot & \cdot & \cdot & |E_n|\cos\theta_n \end{bmatrix} \tag{3.55}$$

The transmission lines of the system can be modeled using a pi-model. The diagram shown in Fig 3.22 represents the pi model of a transmission line connecting nodes k and m with a series impedance $R + jX$ and shunt admittances B_{k0} and B_{m0}, respectively.

The series admittance is given by:

$$Y = \frac{1}{R + jX} = g_{km} + jb_{km} \tag{3.56}$$

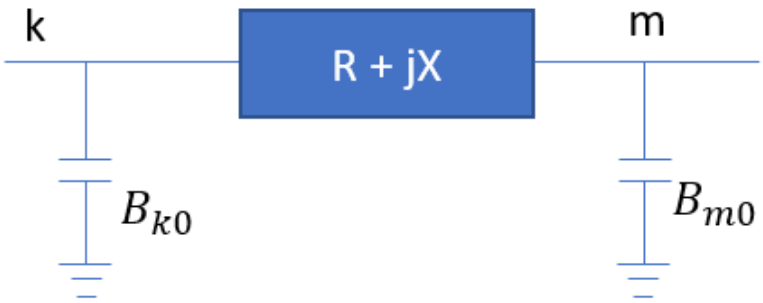

Figure 3.22: Representation of Pi-Model for a Transmission Line

The relationship between the voltages and currents in cartesian form and the voltages in polar form at a node k is given by:

$$\begin{bmatrix} (V_k)_r \\ (V_k)_i \\ (I_{km})_r \\ (I_{km})_i \end{bmatrix} = \begin{bmatrix} |V_k|\cos\theta_k \\ |V_k|\sin\theta_k \\ (|V_k|\cos\theta_k - |V_m|\cos\theta_m)g_{km} - (|V_k|\sin\theta_k - |V_m|\sin\theta_m)b_{km} - b_{k0}|V_k|\sin\theta_k \\ (|V_k|\cos\theta_k - |V_m|\cos\theta_m)b_{km} - (|V_k|\sin\theta_k - |V_m|\sin\theta_m)g_{km} + b_{k0}|V_k|\cos\theta_k \end{bmatrix} \quad (3.57)$$

This equation can be used to obtain the relationship between the current measurements in cartesian form and the states of the system in the cartesian form, which is given by:

$$\begin{bmatrix} I_{(km)_r} \\ I_{(km)_i} \end{bmatrix} = \begin{bmatrix} g_{km} & -g_{km} & -b_{km} - b_{k0} & b_{km} \\ b_{km} + b_{k0} & -b_{km} & b_{km} & -g_{km} \end{bmatrix} * \begin{bmatrix} (V_k)_r \\ (V_m)_r \\ (V_k)_i \\ (V_m)_i \end{bmatrix} \quad (3.58)$$

Similarly, the relationship between the current measurements in cartesian form flowing through a transformer and the system states can also be obtained using the equivalent pi-

model described in Fig 3.8. The matrices $C_{11}, C_{12}, C_{21}, C_{22}$ which are a function of line parameters of the system can be obtained using the above equation.

Since, the matrix H is dependent only on the topology and line parameters and are independent of the system conditions, 3.41 is linear and non-iterative.

Therefore, the linear state estimation formulation involves formulating the Jacobian matrix H and the weight matrix (if the errors are Gaussian). Once the two matrices are formulated, the states can be identified by solving the linear equation for a given set of measurements.

CHAPTER 4

STATE ESTIMATION RESULTS

A neural network-based state estimator requires the identification of the mapping function using historical data that represents the joint probability distribution of the corresponding states of the system for a given measurement set. The procedure for identifying the mapping function and the estimated real-time states using the mapping function for an IEEE 118-bus system was demonstrated in the previous chapter for a system which was redundantly monitored by PMUs. However, the synchrophasor infrastructure is expensive and it is neither practical nor necessary to place PMUs at every bus of the system. Also, the state estimation demonstrated earlier was in the presence of measurements with errors assumed to have come from a Gaussian distribution. This chapter will demonstrate the formulation of state estimation in the presence of limited sensor measurements, different measurement characteristics, varying system conditions, thereby highlighting the advantages of the proposed methodology over the conventional methodologies.

4.1 Estimation with Limited Number of Sensors:

The primary requirement for performing state estimation is data. The major source of data in power systems are the PMUs. The PMUs need not be placed at all buses for the complete observability of the system because of their capabilities to measure voltages and currents. For instance, for an IEEE 14 bus system, 4 PMUs are sufficient to be placed for complete system observability [91] . The number of PMUs placed in the power grid are increasing continuously [92] , however, not enough PMUs are placed on the grid for a complete observability of all voltage levels of the transmission system.

The biggest advantage that a deep learning-based state estimator offers is its ability to identify the mapping between the states and the input measurements. The deep learning-based state estimation performed for the IEEE 118-bus system for the following two configurations of PMU placement are demonstrated below:

1) Minimum number of PMUs placed for complete observability of the system (Complete Observability): The PMU placement optimization problem discussed in Appendix A is solved and it is identified that 32 PMUs placed can provide complete system observability (basic observability with no redundancy). The bus locations of the PMUs for this placement configuration are given in Table 4.1.

Table 4.1: Location of PMUs for Complete Observability

Number of PMUs	**Location of PMUs**
32	3, 5, 9, 12, 15, 17, 21, 25, 29, 34, 37, 40, 45, 52, 56, 62, 64, 68, 70, 71, 76, 78, 85, 86, 89, 92, 96, 100, 105, 110, 114

2) Placement of PMUs at only the highest voltage buses of the system (Partial Observability): There are 11 high voltage buses in the IEEE 118-bus system and the locations of these buses are given by Table 4.2 [93] .

Table 4.2: Location of High Voltage Buses for PMU Placement

Number of PMUs	**Location of PMUs**
11	8, 9, 10, 26, 30, 38, 63, 64, 65, 68, 81

4.1.1 State Estimation in the Presence of Complete Observability:

The first step is to generate the data corresponding to the features and labels of the neural network for the training and validation. The data for the IEEE 118-bus system is generated by solving the ACOPF for the chosen discrete load data points obtained from the generated hourly load pattern. A Gaussian error having parameters mentioned in Table 4.3 are added to create the measurements. The TVE of the error added to the measurements are ensured to be lesser than 1%.

Table 4.3: Parameters of the Gaussian Error Added to the Measurements

Measurement type	Error Mean	Error Standard Deviation
Magnitude (voltages and currents)	0	1/3 %
Angle (voltages and currents)	0	0.573/3°

The testing measurements and the expected outputs are also obtained from the distribution used for creating the training and validation datasets. The measurements will be the complex voltages of the 32 buses where the PMUs are placed, and the complex currents of the branches connected to these buses. The dimensions of the training, validation, and testing data are given below.

Table 4.4: Dimensions of the Different Input Files for Performing State Estimation in Presence of Complete Observability with Normal Data Set

Input file	**Matrix Dimensions**
Training measurements (NN features)	8500*342
Training states (NN labels)	8500*236
Validation measurements (NN features)	2700*342
Validation states (NN labels)	2700*236
Testing measurements (NN new features)	1000*342
Testing states (NN expected output)	1000*236

The same neural network architecture mentioned in Table 3.6 will be used for this application. The number of nodes in the hidden layer was changed to two thirds of the number of inputs added to the total number of outputs, which equals 464.

The neural network was also trained, validated, and tested with a large dataset containing many more samples. The dimensions of the datasets fed to the neural network are mentioned below (Table 4.5). The neural network architecture used for estimating the states of the system had the same hyperparameters as the ones mentioned in Table 3.6. The state estimation was performed, and the values of the different error matrices calculated for the normal dataset case and the large dataset case are shown in Table 4.6 below.

Table 4.5: Dimensions of the Different Input Files for Performing State Estimation in Presence of Complete Observability with Large Data Set

Input file	**Matrix Dimensions**
Training measurements (NN features)	340000*342
Training states (NN labels)	340000*236
Validation measurements (NN features)	90000*342
Validation states (NN labels)	90000*236
Testing measurements (NN new features)	10000*342
Testing states (NN expected output)	10000*236

Table 4.6: Error Metrics of the Estimated Outputs in the Presence of 32 PMUs

Normal data set	**Large data set**
Voltage Magnitude MAE: 0.0010 pu Voltage Angle MAE: 0.1171 degrees Voltage RMSE: 0.0022	Voltage Magnitude MAE: 8.99e-04 pu Voltage Angle MAE: 0.0928 degrees Voltage RMSE: 0.0019

It can be seen again that a size of a data set size similar to a normal data set can accurately estimate the states of the system. Fig 4.1 and Fig 4.2 shown below display bar plot of the calculated Voltage magnitude absolute difference and Voltage angle absolute difference averaged across all samples, respectively, for all the buses in the system. Fig 4.3 and Fig 4.4 shown below displays the true and estimated voltage magnitude and angle of the first testing sample for all buses.

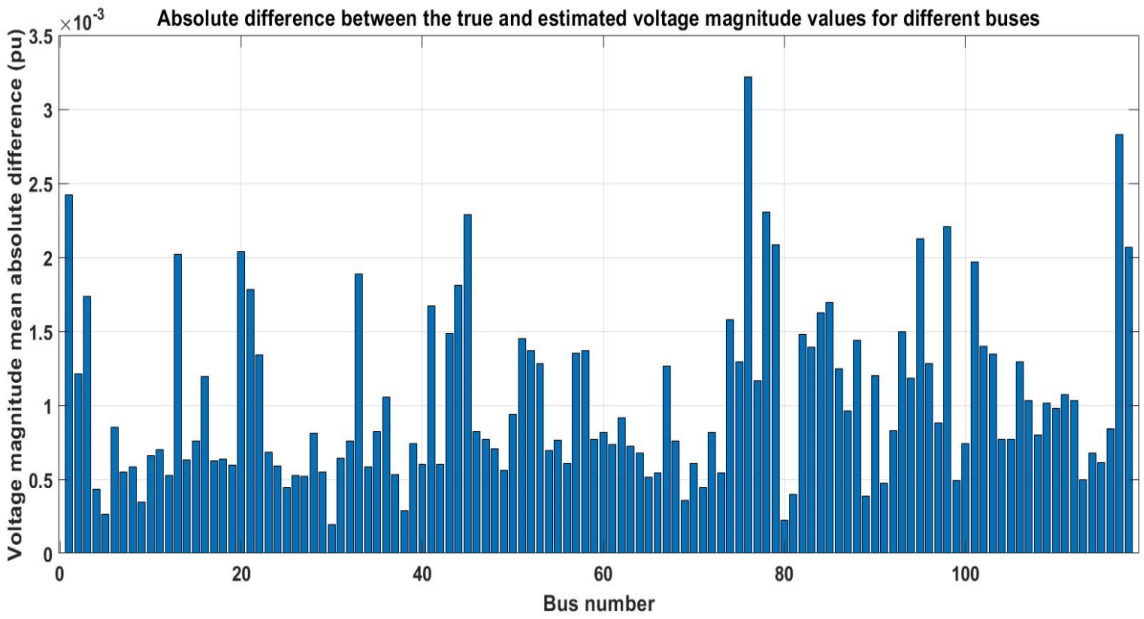

Figure 4.1: Voltage Magnitude Mean Absolute Difference of Estimated and True Values for Different Buses of IEEE 118-Bus System in the Presence of 32 PMUs

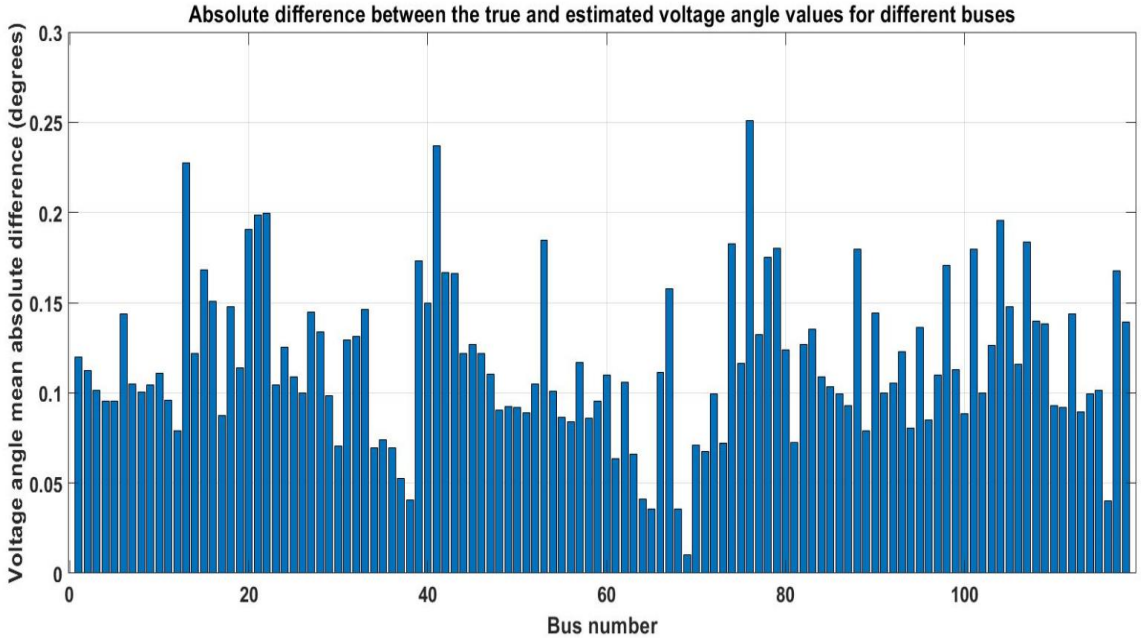

Figure 4.2: Voltage Angle Mean Absolute Difference of Estimated and True Values for Different Buses of IEEE 118-Bus System in the Presence of 32 PMUs

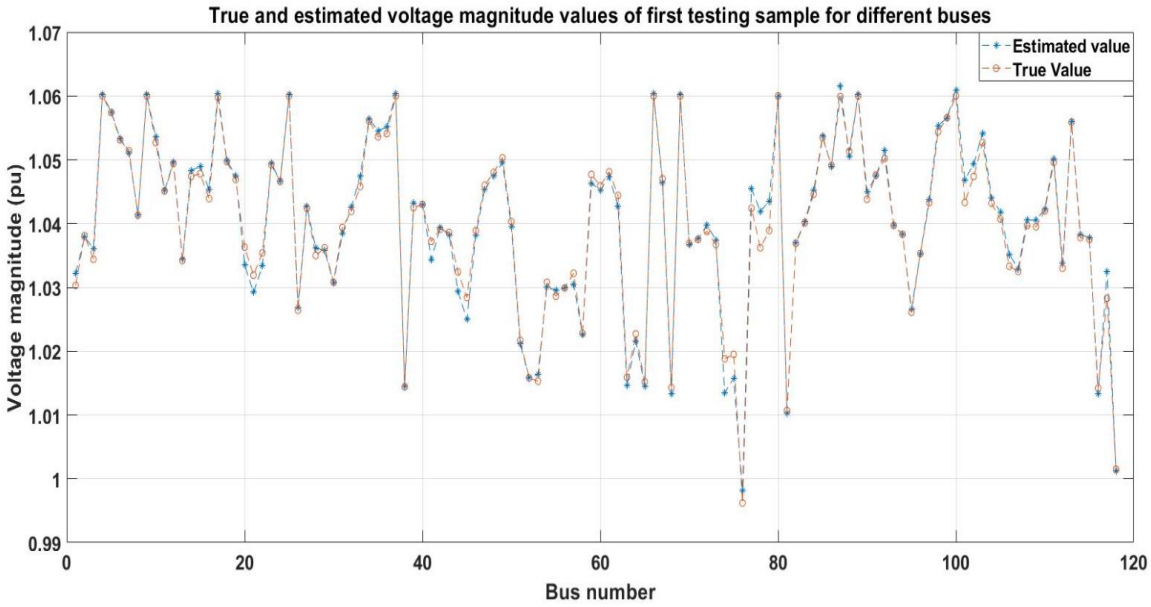

Figure 4.3: Estimated and True Voltage Magnitudes of First Sample for Different Buses of IEEE 118-Bus System in the Presence of 32 PMUs

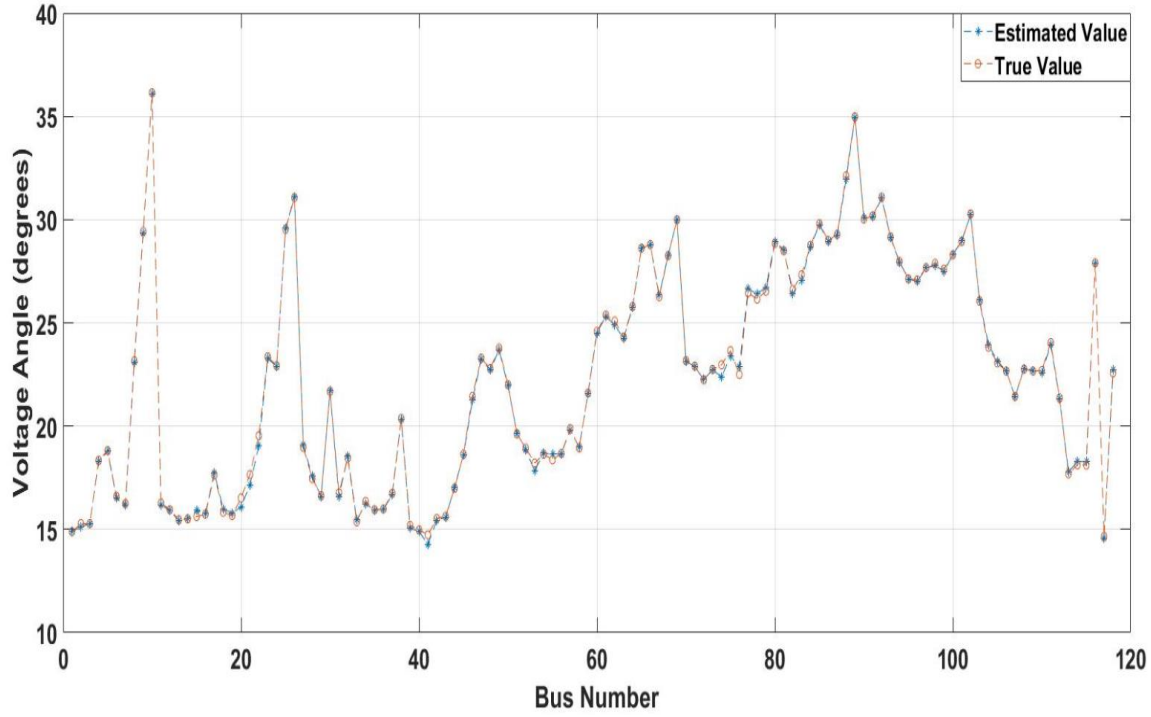

Figure 4.4: Estimated and True Voltage Angles of First Sample for Different Buses in IEEE 118-Bus System in the Presence of 32 PMUs

4.1.2 State Estimation in the Presence of Partial Observability:

The generation of true values from the hourly load distributions and the Gaussian error characteristics used for adding errors to obtain the measurements remains the same as mentioned in Chapter 4.1.1. Furthermore, the state estimation is performed in the presence of a ‘normal data set’ and a ‘large data set’. The dimensions of the different data sets fed into the neural network for both the cases are mentioned below (Tables 4.7 and 4.8).

Table 4.7: Dimensions of the Different Input Files for Performing State Estimation in presence of Partial Observability with Normal Data Set

Input file	**Matrix Dimensions**
Training measurements (NN features)	8500*82
Training states (NN labels)	8500*236
Validation measurements (NN features)	2700*82
Validation states (NN labels)	2700*236
Testing measurements (NN new features)	1000*82
Testing states (NN expected output)	1000*236

The neural network architecture used for the state estimation in the presence of 32 PMUs is used for this application too. The number of nodes in hidden layers alone was changed to 290. The different error metrics calculated for state estimation in the presence of 11 PMUs was tabulated as shown in Table 4.9.

Table 4.8: Dimensions of the Different Input Files for Performing State Estimation in Presence of Partial Observability with Large Data Set

Input file	**Matrix Dimensions**
Training measurements (NN features)	340000*82
Training states (NN labels)	340000*236
Validation measurements (NN features)	90000*82
Validation states (NN labels)	90000*236
Testing measurements (NN new features)	10000*82
Testing states (NN expected output)	10000*236

Table 4.9: Error Metrics of the Estimated Outputs in the Presence of 11 PMUs

Normal data set	**Large data set**
Voltage Magnitude MAE: 0.0013 pu Voltage Angle MAE: 0.2683 degrees Voltage RMSE: 0.0057	Voltage Magnitude MAE: 0.0011 pu Voltage Angle MAE: 0.2419 degrees Voltage RMSE:0.0055

It can be observed that the highlight of the neural network state estimator is its ability to predict the states of the system accurately even if the available set of measurements in the system does not ensure complete topological observability. The bar plot of the absolute voltage magnitude and angle averaged across all samples for all buses in the presence of 11 PMUs is shown in Fig 4.5 and Fig 4.6. The estimated voltage magnitude and angle true values of all buses for the first sample is shown in Fig 4.7 and Fig 4.8.

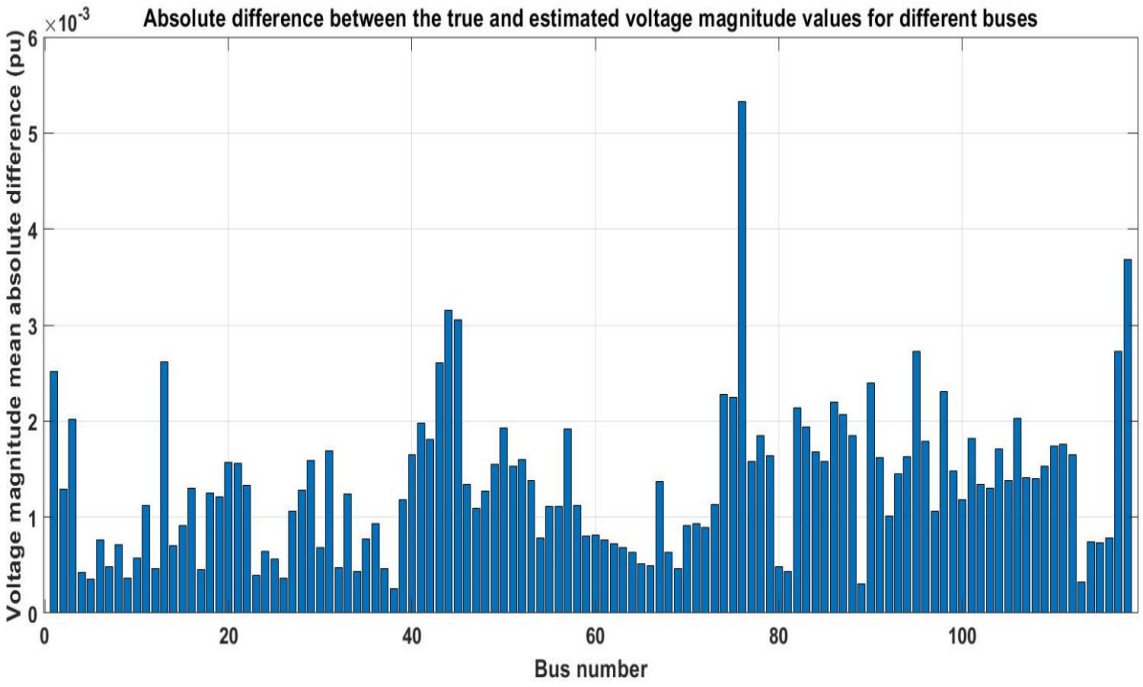

Figure 4.5: Voltage Magnitude Mean Absolute Difference Between Estimated and True Values for Different Buses of IEEE 118-Bus System in the Presence of 11 PMUs

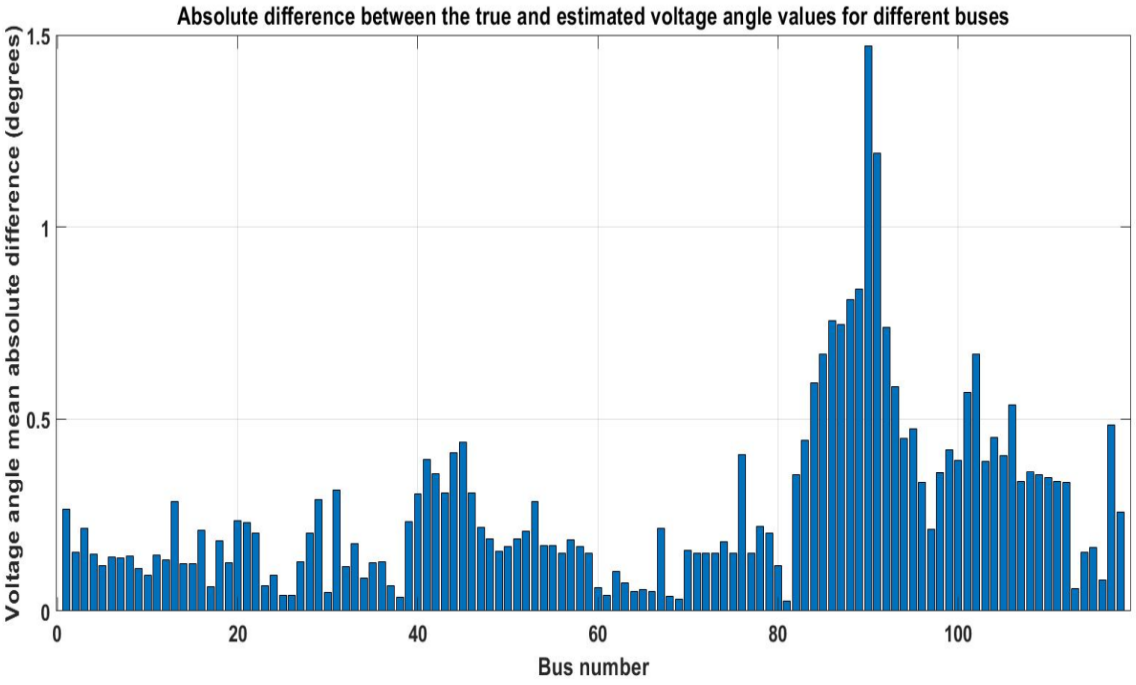

Figure 4.6: Voltage Angle Mean Absolute Difference Between Estimated and True Values for Different Buses of IEEE 118-Bus System in the Presence of 11 PMUs

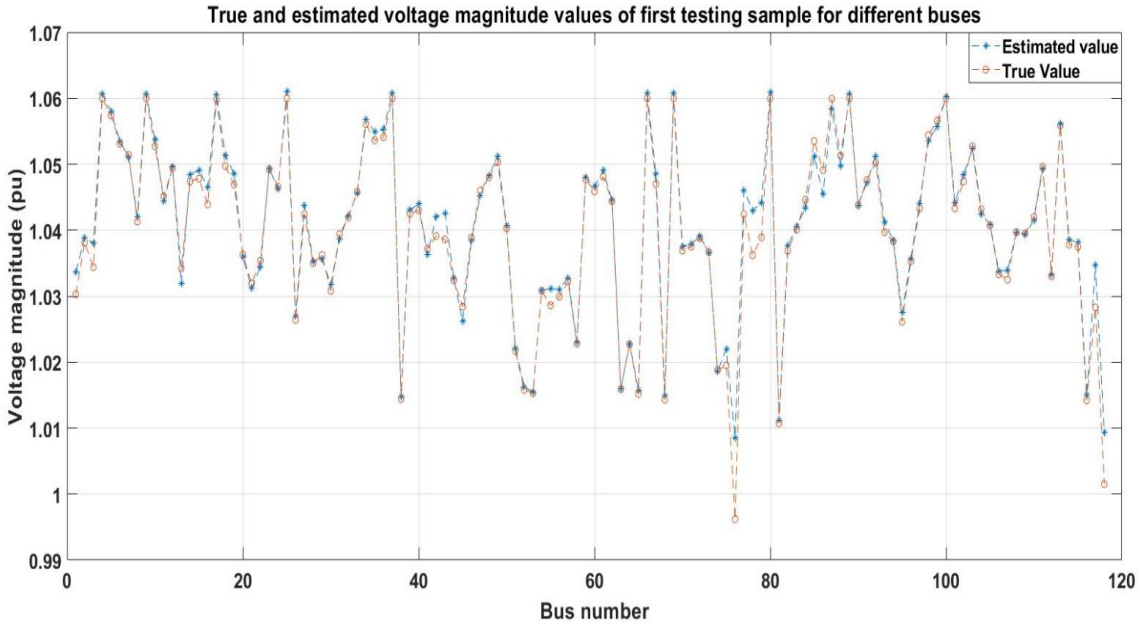

Figure 4.7: Estimated and True Voltage Magnitudes of First Sample for Different Buses of IEEE 118-Bus System in the Presence of 11 PMUs

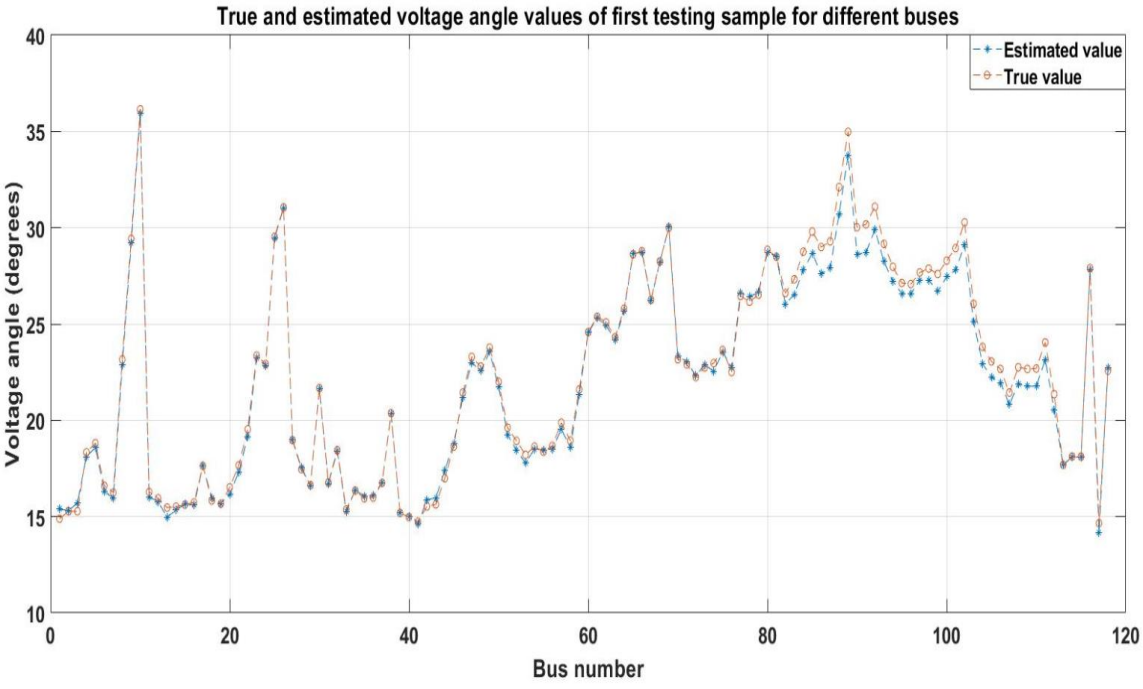

Figure 4.8: Estimated and True Voltage Angles of First Sample for Different Buses of IEEE 118-Bus System in the Presence of 11 PMUs

To summarize, the states of the neural network state estimator were predicted in the presence of redundant measurement set, complete measurement set required for topological observability, and a measurement set not ensuring topological observability of the IEEE 118-bus system. The variation in the error metrics calculated in all three cases is plotted in the graph below in Fig. 4.9 and Fig. 4.10.

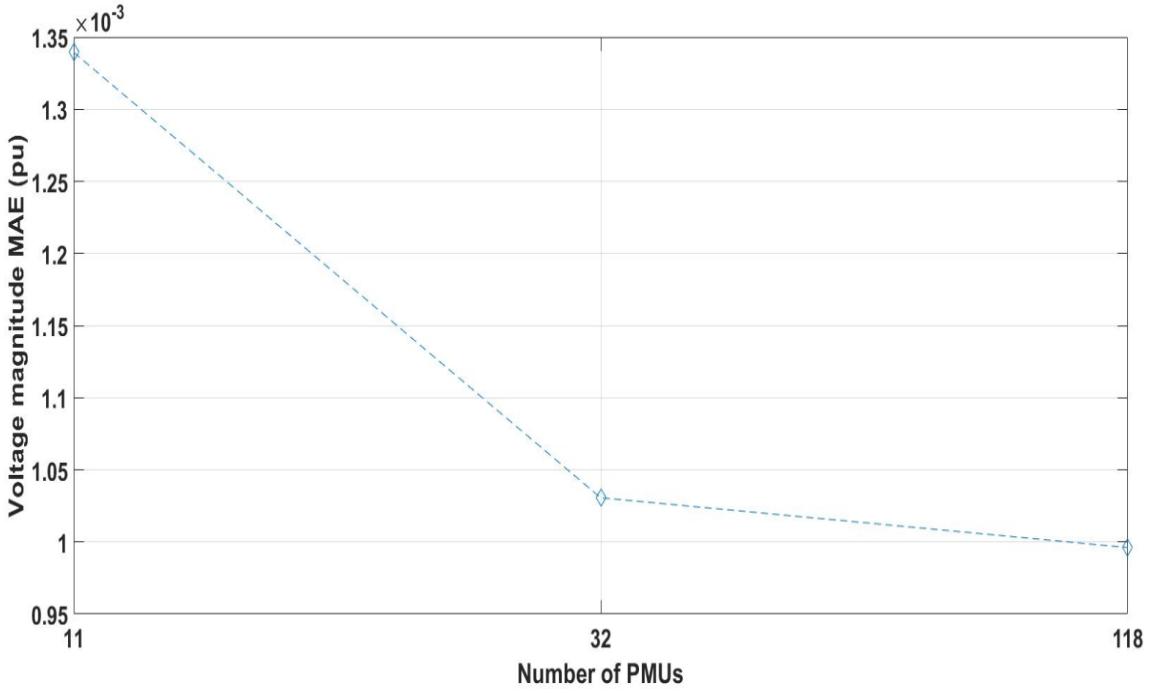

Figure 4.9: Voltage Magnitude MAE of Estimated States for an IEEE 118-Bus System in the Presence of Different Number of PMUs

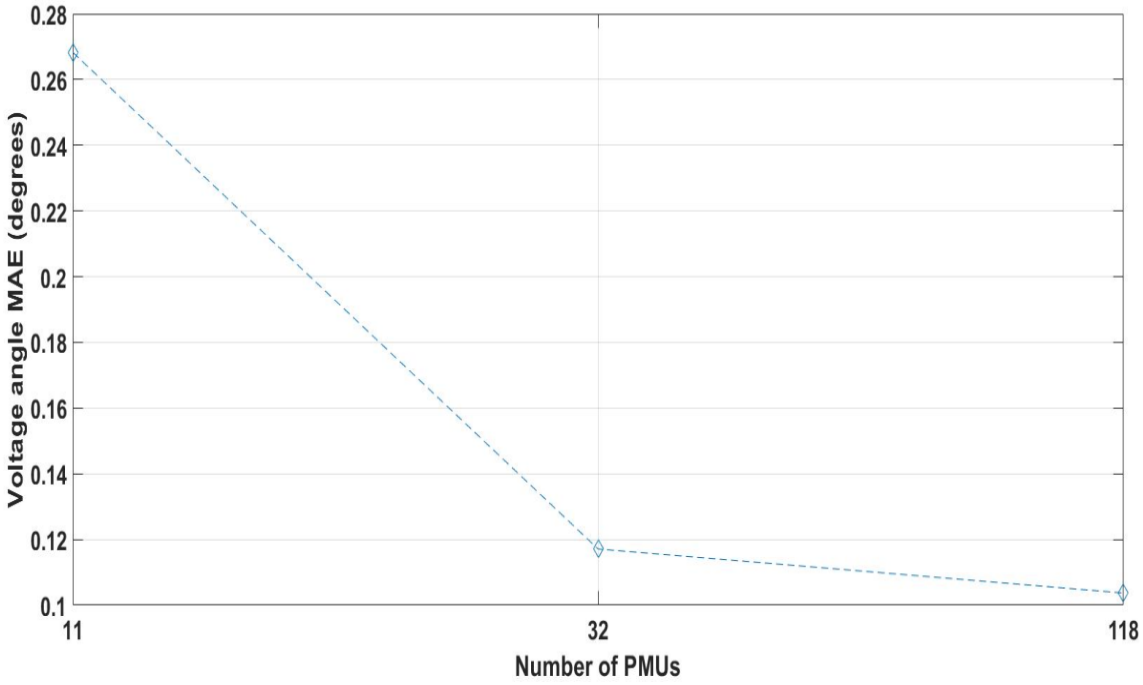

Figure 4.10: Voltage Angle MAE of Estimated States for an IEEE 118-Bus System in the Presence of Different Number of PMUs

4.2 Performance against Linear State Estimator:

The linear state estimator uses a model-based approach, calculating the Jacobian matrix using the topology and line parameter information of the system. This Jacobian matrix represents the relationship between cartesian measurement data and the cartesian states of the system. Additionally, a linear state estimator involves calculation of a Weight matrix, which is a function of the standard deviation of the Gaussian errors present in the measurements.

4.2.1 Error Model of the Synchrophasor Infrastructure:

The errors in the measurements of the PMU can be divided into 2 components:

- Errors in the instrumentation channel: The biggest source of error in the measurements arise as a result of errors in the instrumentation channel before it reaches the PMU device. The instrumentation channel comprises of the voltage transformer, current transformer, burdens, cables, and attenuators [94] .
- Error in the device: The PMU device receives an erroneous input from the instrumentation channel. The device adds an additional random error and the computed phasors by the device will be different from the erroneous input that it received.

The figure below shows the two error components that will be caused by the different components of the synchrophasor infrastructure [95] .

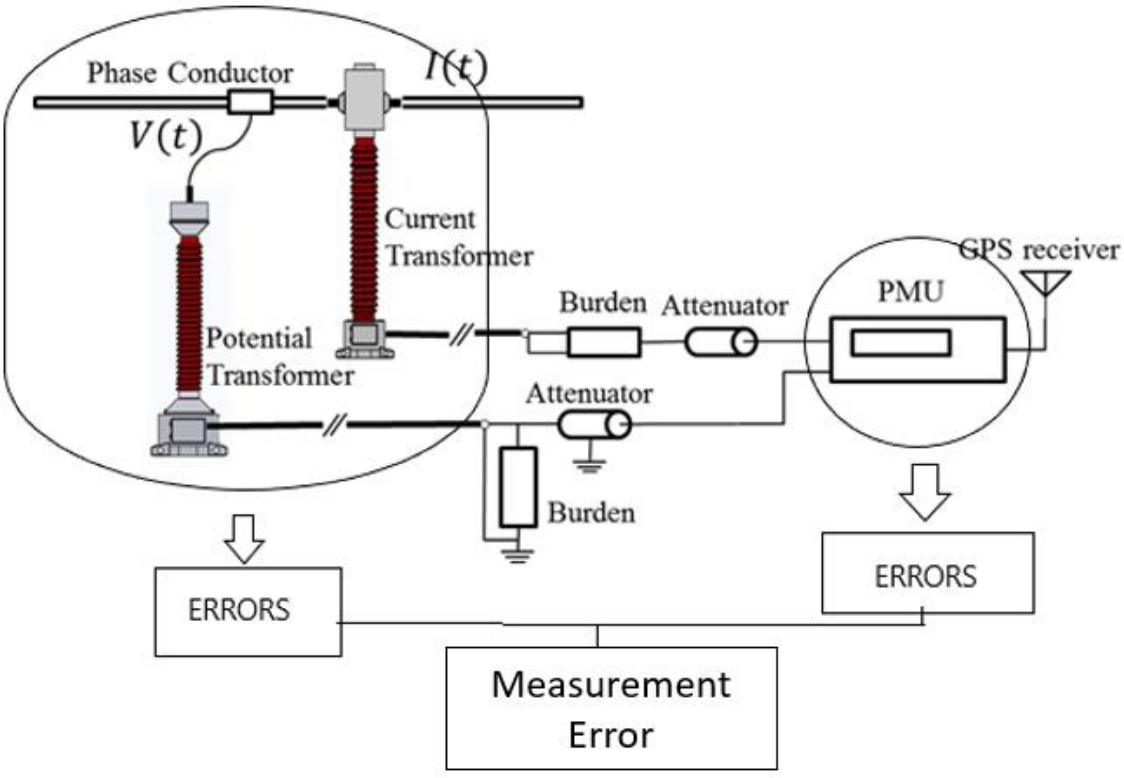

Figure 4.11: Different Error Components of the Synchrophasor Infrastructure

The nature of the instrumentation error was studied in details in [96] and it was shown that the errors of the instrumentation channel is Non-Gaussian and it can be represented best using a 3-component Gaussian Mixture Model (GMM). The minimum and maximum error values that can be introduced by the instrumentation channel on the true voltage and current values mentioned in the IEEE C57.13- 2016 [94] standards are displayed in the table below:

Table 4.10: Error Range of the Instrumentation Channel

Type of Measurement	**Error Range**
Voltage Magnitude	$\pm$ 1.2 %
Voltage Angle	$\pm$ 1°
Current Magnitude	$\pm$ 2.4 %
Current Angle	$\pm$ 2°

The parameters of the 3-component GMM are chosen using values from Table 4.10 in the following manner:

- Mean: Three equidistant points between the minimum and the maximum values are chosen to be the 3 different mean values of the GMM. The mean value of the second component turns out to be 0 for all GMM measurement errors.
- Standard Deviation: If the value σ corresponds to the standard deviation, it is given by:

$$\sigma = \frac{Difference\ between\ the\ 2\ consecutive\ mean\ components}{3} \quad (4.1)$$

The same value of σ is chosen to be the standard deviation for all 3 components.

- Weighing coefficients: A weighing coefficient of components [1/6 2/3 1/6] are chosen for all the GMM measurement errors. Thus, a higher weight is assigned to the errors with a mean component of 0, as the perfectly calibrated PMUs are not expected to have mean errors.

Therefore, the different parameters of the 3 component GMM computed for the voltage magnitude, voltage angle, current magnitude, current angle are mentioned in Tables 4.11-4.14 given below.

Table 4.11: 3-component GMM Parameters of Voltage Magnitude Error

Weighing Coefficients	[1/6 2/3 1/6]
Mean (pu)	[-0.006 0 0.006]
Standard Deviation (pu)	[0.002 0.002 0.002]

Table 4.12: 3-component GMM parameters of Voltage Angle Error

Weighing Coefficients	[1/6 2/3 1/6]
Mean (Degrees)	[-0.5 0 0.5]
Standard Deviation (Degrees)	[0.5/3 0.5/3 0.5/3]

Table 4.13: 3-component GMM parameters of Current Magnitude Error

Weighing Coefficients	[1/6 2/3 1/6]
Mean (pu)	[-0.012 0 0.012]
Standard Deviation (pu)	[0.004 0.004 0.004]

Table 4.14: 3-component GMM parameters of Current Angle Error

Weighing Coefficients	[1/6 2/3 1/6]
Mean (Degrees)	[-1 0 1]
Standard Deviation (Degrees)	[1/3 1/3 1/3]

The error values are picked randomly from this distribution and the erroneous values fed to the PMU device as the input are obtained using the equations mentioned below:

$$v_{em} = v_{tm} * (1 + e1_{vm}) \tag{4.2}$$

$$v_{ea} = v_{ta} + e1_{va} \tag{4.3}$$

$$c_{em} = c_{tm} * (1 + e1_{cm}) \tag{4.4}$$

$$c_{ea} = c_{ta} + e1_{ca} \tag{4.5}$$

The values v_{em}, v_{ea}, c_{em}, c_{ea} corresponds to the erroneous voltage magnitudes, voltage angles, current magnitudes and current angles fed to the PMU device and the values v_{tm}, v_{ta}, c_{tm}, c_{ta} refer to the corresponding true values. The values $e1_{vm}$, $e1_{va}$, $e1_{cm}$, $e1_{ca}$ that are added to the true values refer to the error values chosen from the 3-component GMM.

The error component due to the device is modeled as a Gaussian error with the error parameters mentioned in the table below.

Table 4.15: Parameters of the Gaussian Error

Measurement type	Error Mean	Error Standard Deviation
Magnitude (voltages and currents)	0	1/3 %
Angle (voltages and currents)	0	0.573/3°

The error values of Table 4.15 are chosen in a manner that the TVE was less than or equal to 1%. If $e2_{vm}, e2_{va}, e2_{cm}, e2_{ca}$ correspond to the error values in the voltage magnitudes, voltage angles, current magnitudes and current angles chosen from the Gaussian distribution, the output measurements of the PMU v_m, v_a, c_m, c_a are given by:

$$v_m = v_{em} * (1 + e2_{vm}) \tag{4.6}$$

$$v_a = v_{ea} + e2_{va} \tag{4.7}$$

$$c_m = c_{em} * (1 + e2_{cm}) \tag{4.8}$$

$$c_a = c_{ea} + e2_{ca} \tag{4.9}$$

The neural network state estimator is now formulated in the presence of measurements from 32 PMUs, but by adding errors from a 3 component GMM and a Gaussian distribution. The training, validation and testing outputs would be the true values of the complex voltages for all the buses. This neural network-based state estimator has the same architecture as the one mentioned in Table 3.6.

Similarly, a linear state estimator is also formulated in the presence of 32 PMUs using measurements obtained by adding a 3 component GMM and Gaussian errors to the true values. The voltage magnitude, angle mean absolute differences obtained using the neural network state estimator and the linear state estimator in the presence of 32 PMUs for different buses are plotted in Figs. 4.12-4.15.

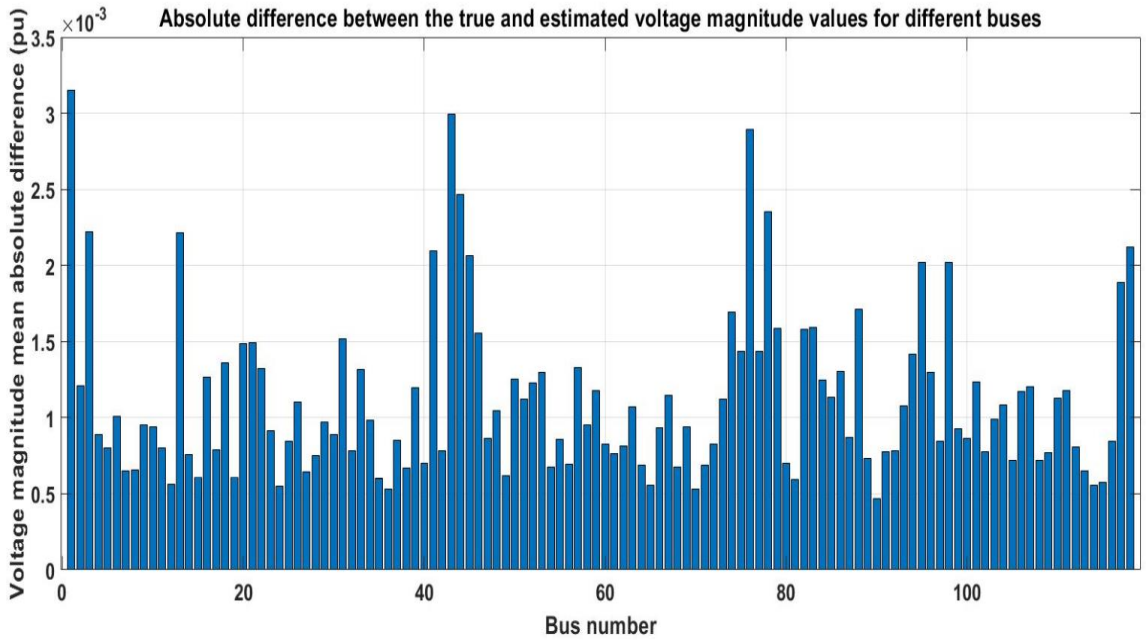

Fig 4.12: Voltage Magnitude Mean Absolute Difference Plot for Neural Network State Estimator with Errors from GMM and Gaussian Distribution in the Presence of 32 PMUs

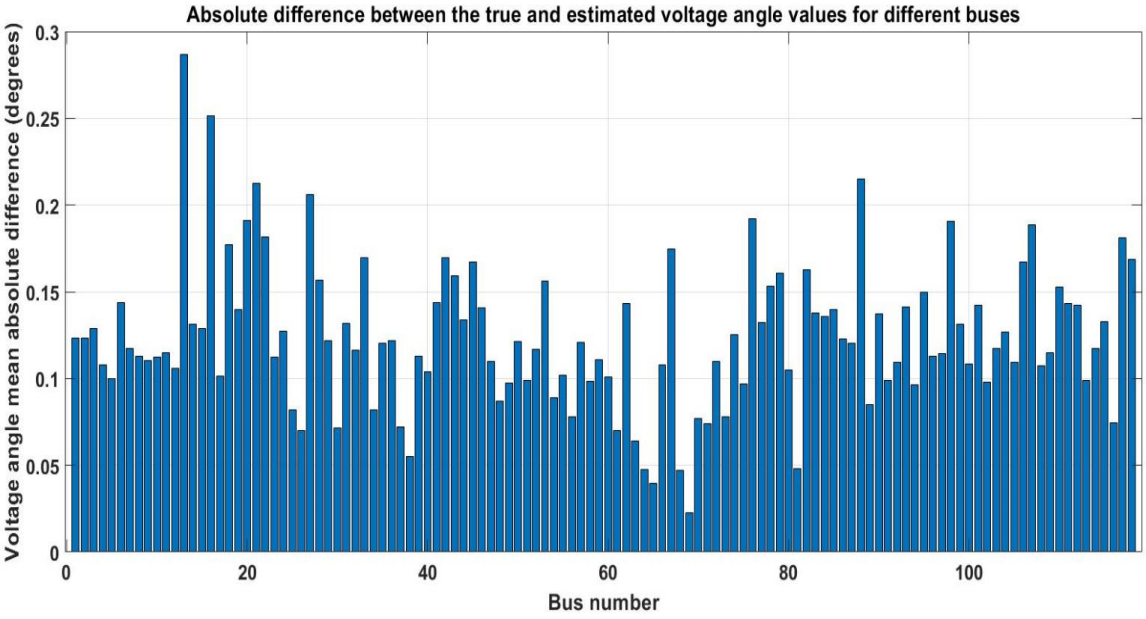

Fig 4.13: Voltage Angle Mean Absolute Difference Plot for Neural Network State Estimator with Errors from GMM and Gaussian Distribution in the Presence of 32 PMUs

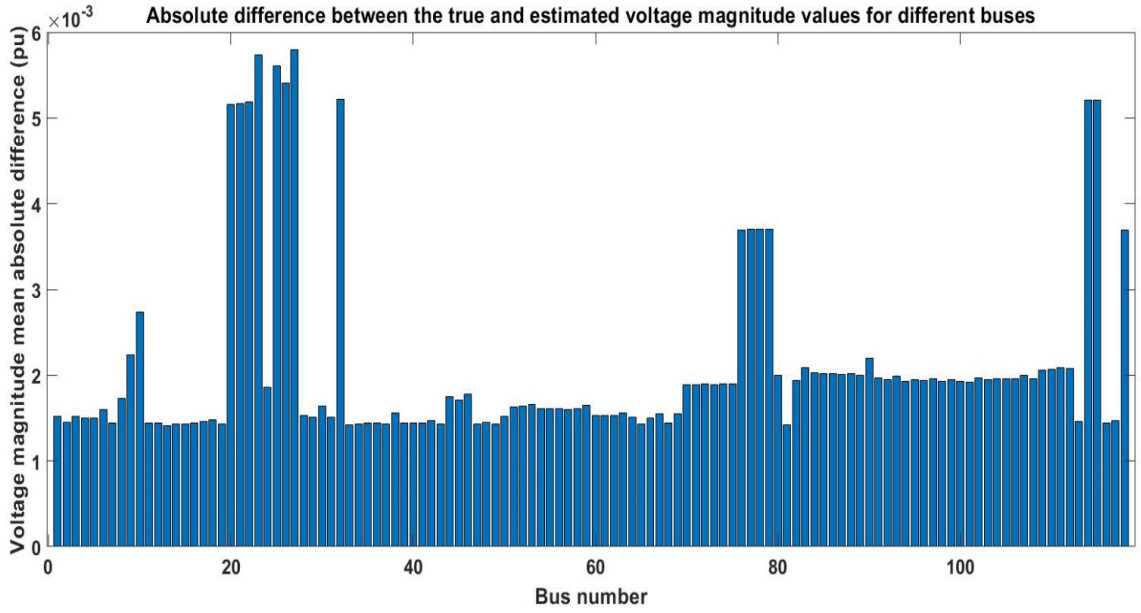

Fig 4.14: Voltage Magnitude Mean Absolute Difference Plot for Linear State Estimator with Errors from GMM and Gaussian Distribution in the Presence of 32 PMUs

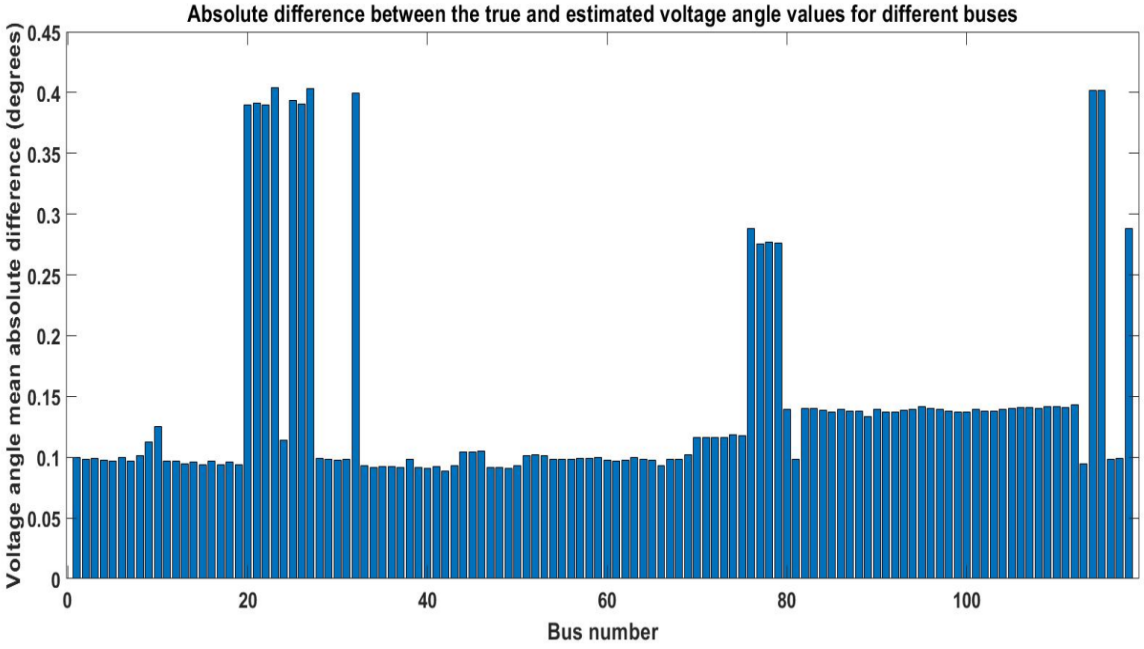

Fig 4.15: Voltage Angle Mean Absolute Difference Plot for Linear State Estimator with Errors from GMM and Gaussian distribution in the Presence of 32 PMUs

The values of different metrics obtained using both the techniques are displayed in Table 4.16 below.

Table 4.16: Error Metrics of Estimated States with GMM and Gaussian Error in Measurements in the Presence of 32 PMUs

NN State Estimator	**Linear State Estimator**
Voltage Magnitude MAE: 0.0011 pu Voltage Angle MAE: 0.1194 degrees Voltage RMSE: 0.0022	Voltage Magnitude MAE: 0.0021 pu Voltage Angle MAE: 0.1431 degrees Voltage RMSE: 0.0033

It can be observed that a neural network state estimator performs better than a linear state estimator in the presence of a 3-component GMM and Gaussian errors in a system even when it is completely observed by PMUs.

Now, a neural network state estimator is formulated in the presence of measurements from 11 PMUs. The training, validation and testing features for the neural network are obtained for a normal data set in the presence of 11 PMUs by adding errors from a 3 component GMM and a Gaussian distribution. The training, validation, and testing outputs would be the true values of the complex voltages for all the buses. This neural network based state estimator also has the same architecture as the one mentioned in Table 3.6 and Chapter 4.1.

The plots of the voltage magnitude and angle absolute differences obtained are shown in Fig. 4.16 and Fig. 4.17.

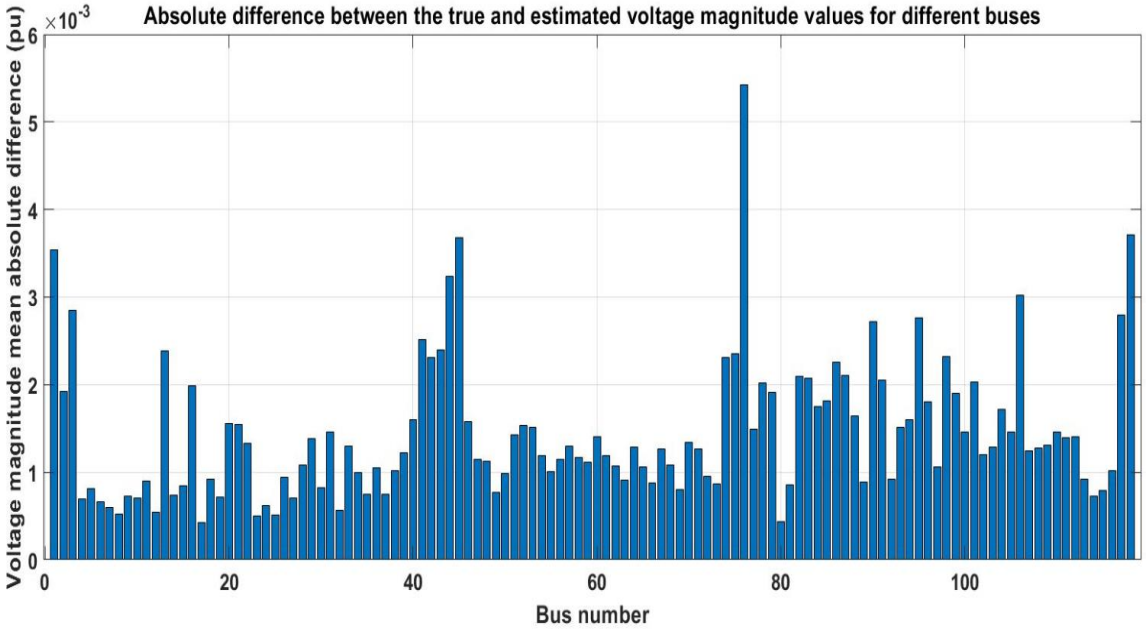

Fig 4.16: Voltage Magnitude Mean Absolute Difference Plot for Neural Network State Estimator with Errors from GMM and Gaussian Distribution in the Presence of 11 PMUs

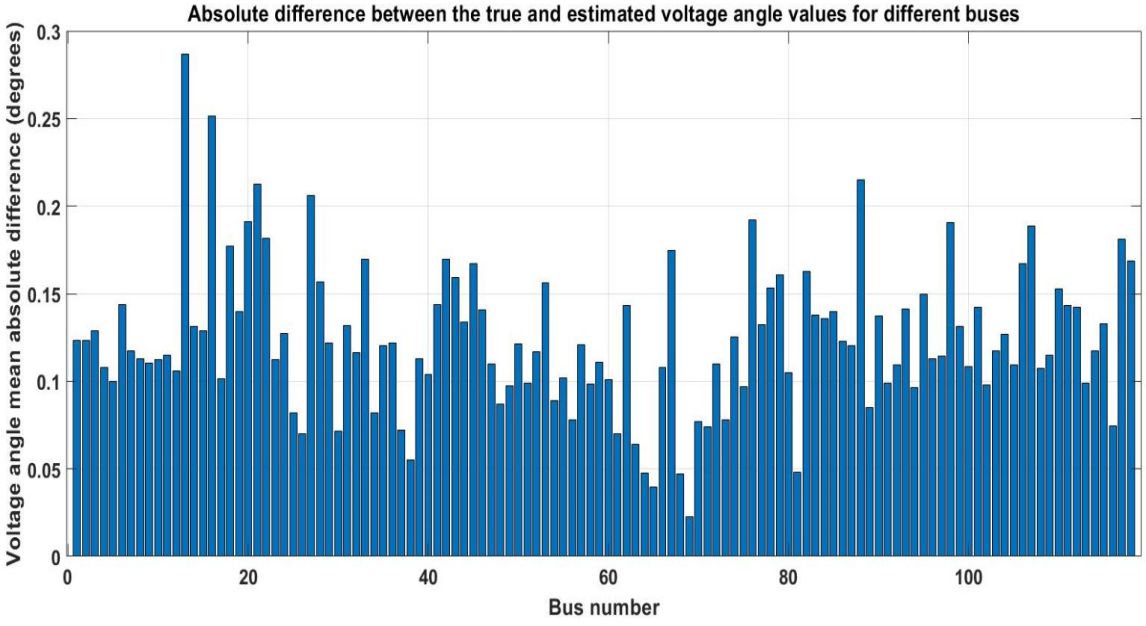

Fig 4.17: Voltage Angle Mean Absolute Difference Plot for Neural Network State Estimator with Errors from GMM and Gaussian Distribution in the Presence of 11 PMUs

The error metrics obtained after performing state estimation are displayed in Table 4.17.

Table 4.17: Error Metrics of Estimated States with GMM and Gaussian Error in Measurements in the Presence of 11 PMUs

Voltage Magnitude MAE: 0.0014 pu
Voltage Angle MAE: 0.2829 degrees
Voltage RMSE: 0.0058

The biggest advantage of a neural network state estimator over LSE is its ability to perform an accurate state estimation in the presence of 3 component GMM errors and Gaussian errors even when the system is incompletely observed by PMUs. In contrast, the linear state estimator will not be able to estimate the states of all buses when PMUs do not completely observe the system.

4.2.2 State Estimation in Presence of Laplacian Errors:

The combined effect of the errors due to the instrumentation channel and the device results in the errors of the synchrophasor infrastructure to be Non-Gaussian. The two commonly used non-Gaussian error models are the Gaussian Mixture Models and the thick tailed Laplacian noise models. The performance of the neural network state estimator and the linear state estimator using a 3 component GMM was discussed in Chapter 4.2.1; in this section, we will analyze their performance in presence of Laplacian noise in the measurements.

The PDF of a Laplacian distribution characterized by the location parameter (μ) and the scaling parameter (b) for a value x is given by:

$$f(x\,|\mu, b) = \frac{1}{2b}\, e^{\left(-\frac{|x-\mu|}{b}\right)} \tag{4.10}$$

$$f(x\,|\mu, b) = \frac{1}{2b}\begin{cases} e^{\left(-\frac{|x-\mu|}{b}\right)} & if\ x \geq \mu \\ e^{\left(-\frac{|\mu-x|}{b}\right)} & if\ x \geq \mu \end{cases} \tag{4.11}$$

The PDF corresponding to a Laplacian distribution was generated by using a location parameter of 0 and scaling parameter of 0.001 as mentioned in [97] . The errors for the measurements were chosen randomly from the generated Laplacian distribution PDF.

A neural network state estimator is now formulated in the presence of measurements from 32 PMUs by adding errors from the Laplacian distribution. The training, validation, and testing outputs would be the true values of the complex voltages for all the buses. This neural network-based state estimator has the same architecture as the one mentioned in Table 3.6 and explained in Chapter 4.1.

A linear state estimator is also formulated with measurement errors from Laplacian distribution in the presence of 32 PMUs. The voltage magnitude and angle mean absolute differences observed in case of both the state estimators for all the buses is shown in Figs. 4.18-4.21.

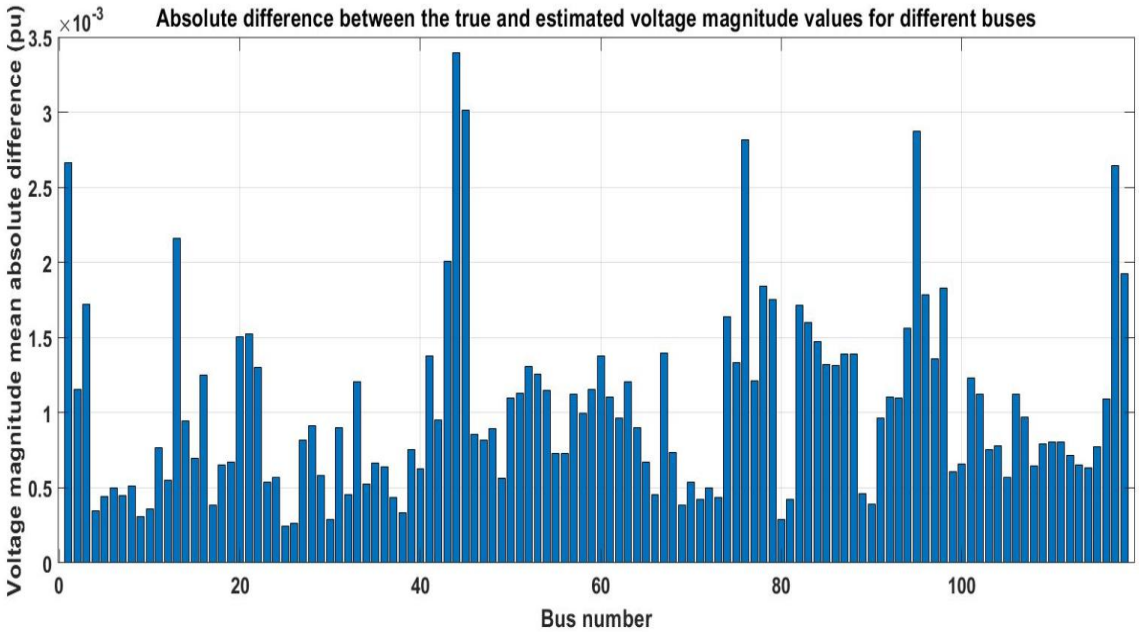

Fig 4.18: Voltage Magnitude Mean Absolute Difference Plot for Neural Network State Estimator with Errors from Laplacian Distribution in the Presence of 32 PMUs

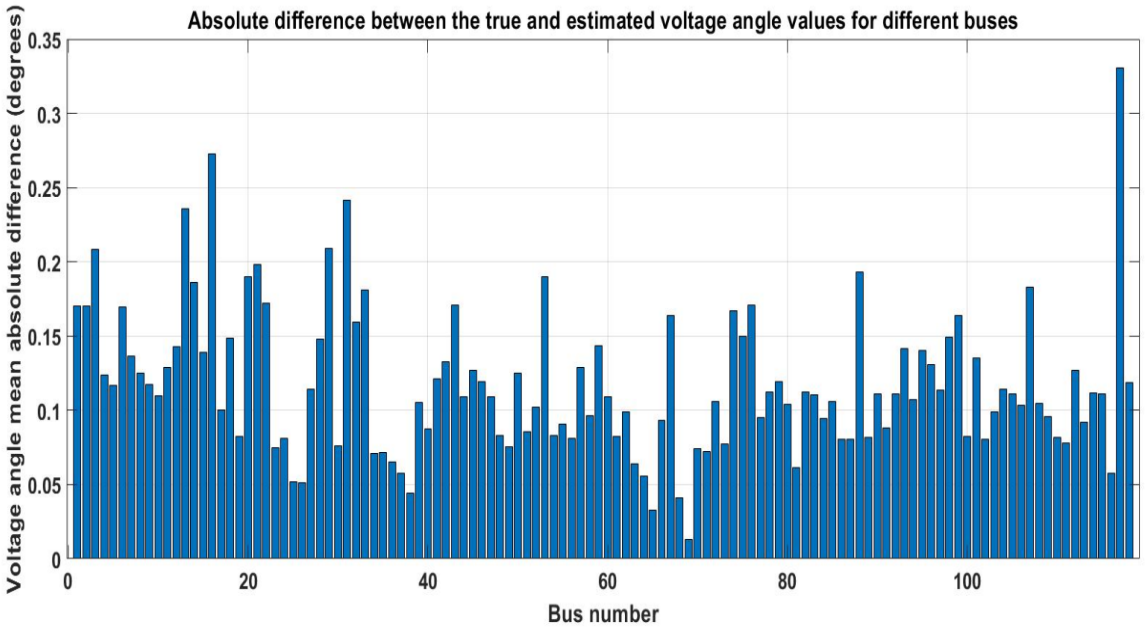

Fig 4.19: Voltage Angle Mean Absolute Difference Plot for Neural Network State Estimator with Errors from Laplacian Distribution in the Presence of 32 PMUs

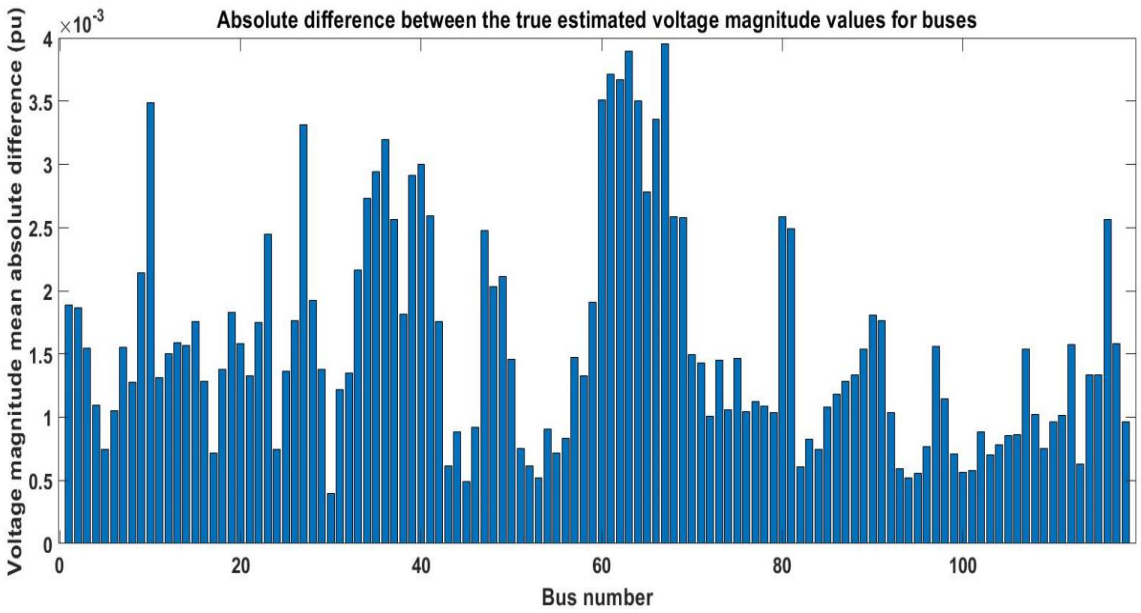

Fig 4.20: Voltage Magnitude Mean Absolute Difference Plot for Linear State Estimator with Errors from Laplacian Distribution in the Presence of 32 PMUs

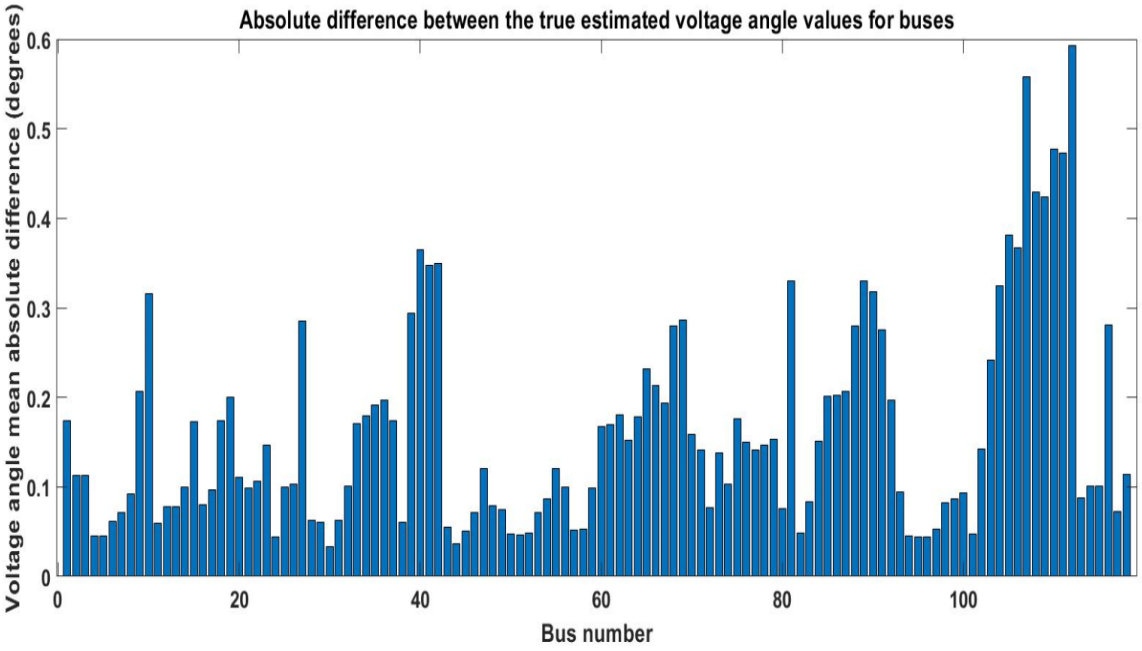

Fig 4.21: Voltage Angle Mean Absolute Difference Plot for Linear State Estimator with Errors from Laplacian Distribution in the Presence of 32 PMUs

The values of different metrics obtained using both the techniques are displayed in Table 4.18 below.

Table 4.18: Error Metrics of Estimated States with Error in Measurements from a Laplacian Distribution in the presence of 32 PMUs

NN State Estimator	**Linear State Estimator**
Voltage Magnitude MAE: 0.0010 pu Voltage Angle MAE: 0.1178 degrees Voltage RMSE: 0.0022	Voltage Magnitude MAE: 0.0016 pu Voltage Angle MAE: 0.1613 degrees Voltage RMSE: 0.0029

It can be observed that a Neural Network State Estimator performs better than a Linear State Estimator in the presence of Laplacian errors even in a completely observed system.

Now, a neural network state estimator is formulated with help of measurements from 11 PMUs placed at high voltage buses. The measurements for the neural network training, validation, and testing are obtained by adding errors to the true values, sampled from Laplacian distribution parameter values mentioned earlier. The states corresponding to the measurements of the training and validation data are the true voltages of the system. This neural network-based state estimator has the same architecture as the one mentioned in Table 3.6 and Chapter 4.1. The plot of the voltage magnitude and angle mean absolute difference is shown below in Figs. 4.22-4.23.

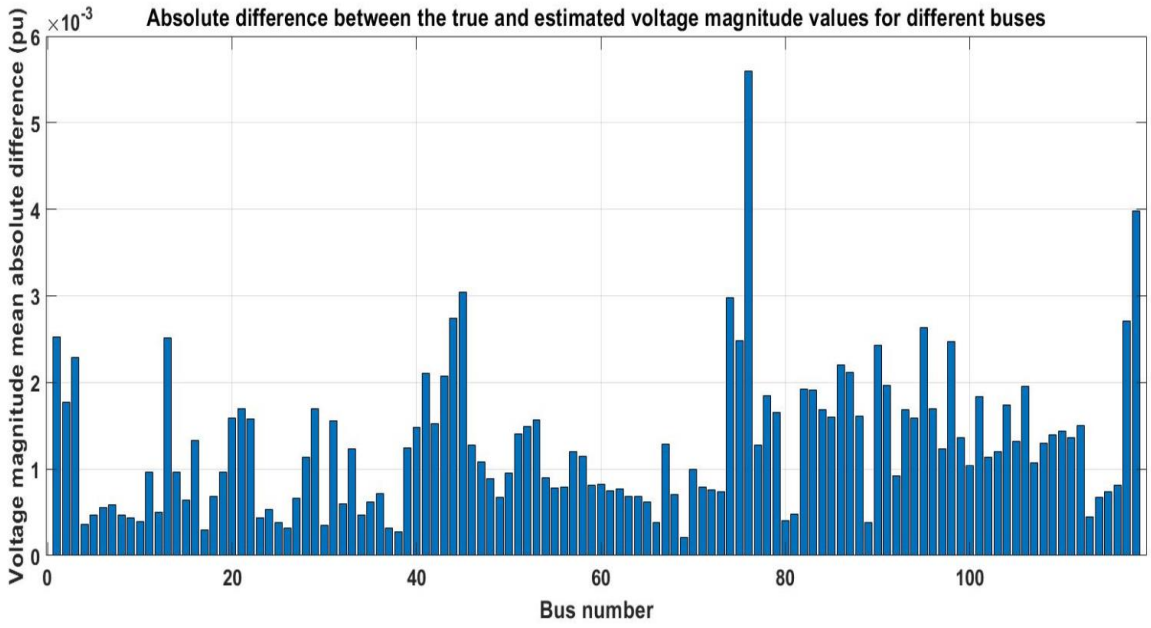

Fig 4.22: Voltage Magnitude Mean Absolute Difference Plot for Neural Network State Estimator with Errors from Laplacian Distribution in the Presence of 11 PMUs

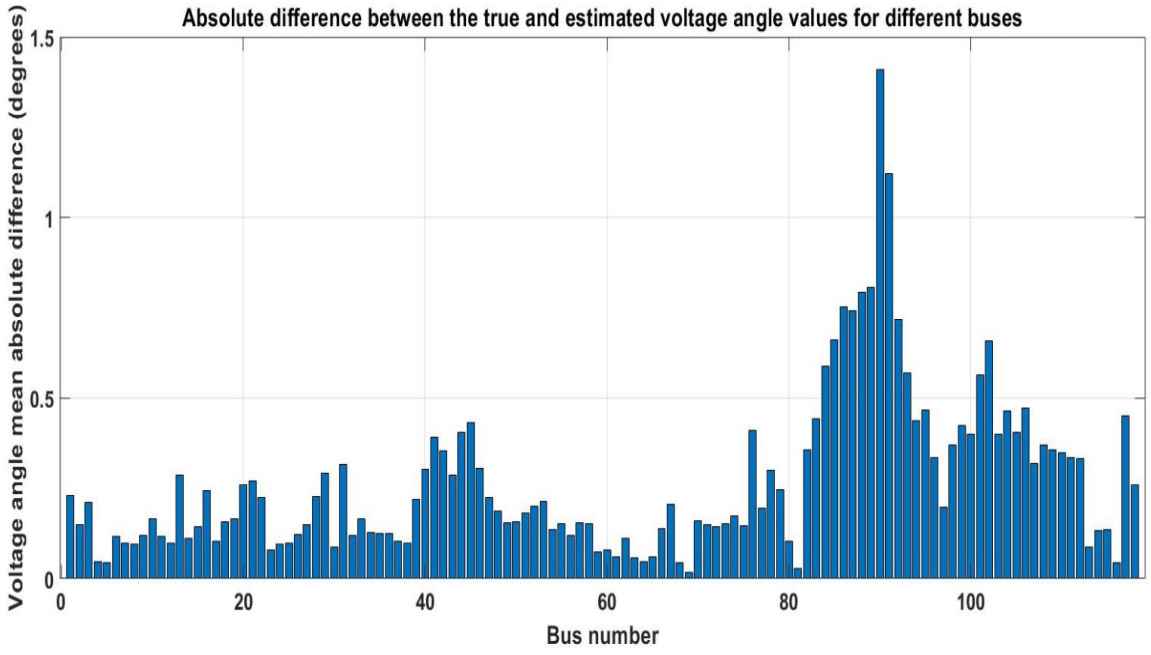

Fig 4.23: Voltage Angle Mean Absolute Difference Plot for Neural Network State Estimator with Errors from Laplacian distribution in the Presence of 11 PMUs

The error metrics obtained after performing the state estimation are given by:

Table 4.19: Error Metrics of Estimated States with Laplacian Error in Measurements in the Presence of 11 PMUs

Voltage Magnitude MAE: 0.0013 pu
Voltage Angle MAE: 0.2659 degrees
Voltage RMSE: 0.0056

The results obtained above prove that the neural network-based state estimator formulated in presence of errors in measurements from Laplacian distribution can estimate the states accurately for a topologically unobserved system. A linear state estimator cannot estimate the states of all the buses for a topologically unobservable system.

4.3 Robustness of the State Estimator

The formulation of state estimator primarily depends on the measurements and the model of the power system. Even though the neural network state estimation formulation is independent of the physical power system model, the ACOPF used to obtain the data during the offline training and validation uses the topology and line parameter information of the system. The line parameters of the transmission system are dependent on the external environmental conditions and vary dynamically [98] . Also, it was earlier mentioned that the characteristics of error in the measurements are non-Gaussian. The performance of the state estimator in the presence of different error characteristics and varying line parameters will be demonstrated below.

4.3.1 Performance of the state estimator against different characteristics of error in the measurements:

The true values of the voltages and currents of the IEEE 118-bus system were obtained from the ACOPF solution of the discrete load data points obtained from the load patterns of the synthetic Texas system. The state estimator was formulated using the architectures explained in Table 3.6 and Chapter 4.1. The performance to the state estimator was analyzed using the measurements obtained by adding errors from the Gaussian distributions, a combination of 3-component GMM and Gaussian distributions and Laplacian distributions. The error metrics obtained for the cases with measurement generation from 3 different distributions are tabulated for the systems with redundant observability (118 PMUs), complete observability (32 PMUs), and partial observability (11 PMUs) in Tables 4.20-4.22.

Table 4.20: Error Metrics of Estimated States using Measurements with Different Error Distributions in the Presence of 118 PMUs

Errors from Gaussian distribution	**Errors from 3-component GMM and Gaussian distribution**	**Errors from Laplacian distribution**
Voltage Magnitude MAE: 9.9609e-04 pu Voltage Angle MAE: 0.1037 degrees Voltage RMSE: 0.0020	Voltage Magnitude MAE: 9.9987e-04 pu Voltage Angle MAE: 0.1077 degrees Voltage RMSE: 0.0022	Voltage Magnitude MAE: 9.3479e-04 pu Voltage Angle MAE: 0.1062 degrees Voltage RMSE: 0.0020

Table 4.21: Error Metrics of Estimated States using Measurements with Different Error Distributions in the Presence of 32 PMUs

Errors from Gaussian distribution	**Errors from 3-component GMM and Gaussian distribution**	**Errors from Laplacian distribution**
Voltage Magnitude MAE: 0.0010 pu Voltage Angle MAE: 0.1171 degrees Voltage RMSE: 0.0022	Voltage Magnitude MAE: 0.0011 pu Voltage Angle MAE: 0.1194 degrees Voltage RMSE: 0.0022	Voltage Magnitude MAE: 0.0010 pu Voltage Angle MAE: 0.1178 degrees Voltage RMSE: 0.0022

Table 4.22: Error Metrics of Estimated States using Measurements with Different Error Distributions in the Presence of 11 PMUs

Errors from Gaussian distribution	**Errors from 3-component GMM and Gaussian distribution**	**Errors from Laplacian distribution**
Voltage Magnitude MAE: 0.0013 pu Voltage Angle MAE: 0.2683 degrees Voltage RMSE: 0.0057	Voltage Magnitude MAE: 0.0014 pu Voltage Angle MAE: 0.2829 degrees Voltage RMSE: 0.0058	Voltage Magnitude MAE: 0.0013 pu Voltage Angle MAE: 0.2659 degrees Voltage RMSE: 0.0056

It can be observed that the estimates have similar ranges of error for the measurements from the 3 different error distributions. This trend of estimates is also observed to be true

for the systems with different number of PMUs installed. Therefore, it can be concluded that the proposed neural network state estimator is robust to the distributions followed by the errors in the measurements.

4.3.2 Performance of the state estimator against varying line parameters:

The line parameters vary dynamically and the exact values of the line parameters in real-time are often unknown. Therefore, the line parameter values using which the neural network was trained and validated would be different from the line parameters in real-time. The following steps describe the process of obtaining the data to be fed into the neural networks while taking this phenomenon into account:

- The training, validation, and test data sets are obtained from the hourly load pattern distributions.
- The ACOPF used to obtain true values from discrete load data points is solved using the default line parameter values for training and validation data and a different line parameter value is used for the testing data.
- The figure below mentioned in [99] displays the dynamic variation of the zero-sequence resistance and reactances of an actual transmission line.

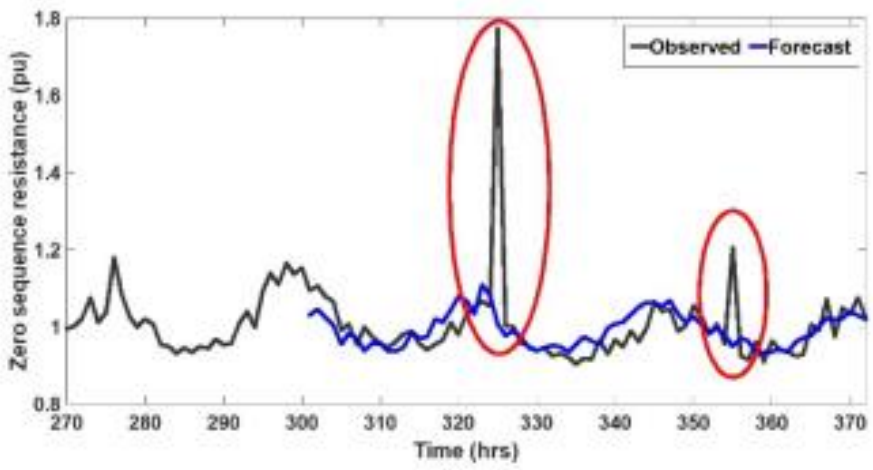

Figure 4.24: Dynamic Variation of Zero Sequence Resistance of a Transmission Line

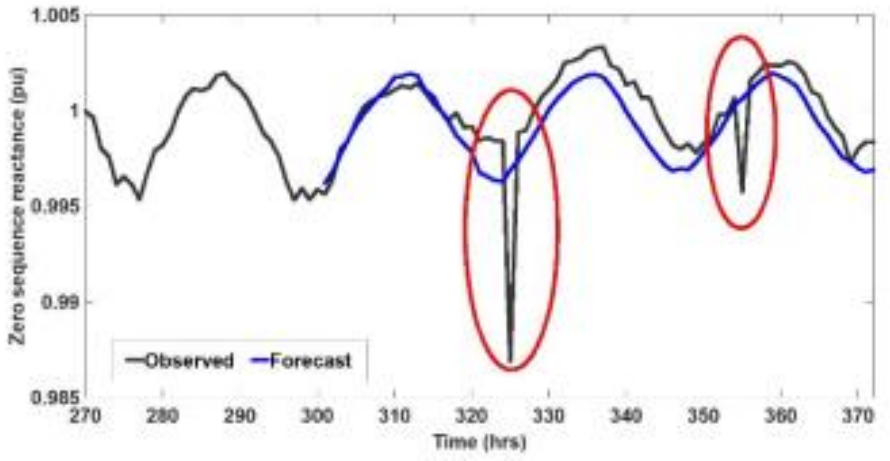

Figure 4.25: Dynamic Variation of Zero Sequence Reactance of a Transmission Line

- The state estimates will be the least accurate when the deviation from the original value of line parameters is high. Therefore, the ACOPF for obtaining testing data set is performed using parameter values of 120% of original resistance and 99.5% of original reactance. The original resistance and reactance correspond to the line parameter values used during training and validation.

- The true values of currents for training, validation, and testing are now generated with the line parameter values used in running ACOPF, respectively.
- A Gaussian error of parameters mentioned in Table 4.15, ensuring lesser than 1% TVE is added to the true values to obtain the measurements. Similarly, a combination of 3-component GMM and Gaussian explained in Chapter 4.2 is also added to the true values to obtain another set of measurements.
- The neural network architecture and dimensions of input features to the neural network for different sensor configurations remain the same as explained in Chapter 3 and Chapter 4.1.

The error metrics obtained by performing a neural network state estimation for changed line parameters and unchanged line parameters for a system with redundant observability, complete observability, and incomplete observability are displayed below in Tables 4.23-4.25.

Table 4.23: Error Metrics of the Estimated States in the Presence of 118 PMUs

NN State Estimator for unchanged line parameters with Gaussian errors	**NN State Estimator for a change in line parameters with Gaussian errors**	**NN State Estimator for a change in line parameters with GMM and Gaussian errors**
Voltage Magnitude MAE: 9.9609e-04 pu	Voltage Magnitude MAE: 0.0037 pu	Voltage Magnitude MAE: 0.0036 pu

Voltage Angle MAE: 0.1037 degrees Voltage RMSE: 0.0020	Voltage Angle MAE: 0.1422 degrees Voltage RMSE: 0.0039	Voltage Angle MAE: 0.2152 degrees Voltage RMSE: 0.0039

Table 4.24: Error Metrics of the Estimated States in the Presence of 32 PMUs

NN State Estimator for unchanged line parameters with Gaussian errors	**NN State Estimator for a change in line parameters with Gaussian errors**	**NN State Estimator for a change in line parameters with GMM and Gaussian errors**
Voltage Magnitude MAE: 0.0010 pu Voltage Angle MAE: 0.1171 degrees Voltage RMSE: 0.0022	Voltage Magnitude MAE: 0.0046 pu Voltage Angle MAE: 0.1305 degrees Voltage RMSE: 0.0043	Voltage Magnitude MAE: 0.0041 pu Voltage Angle MAE: 0.1437 degrees Voltage RMSE: 0.0042

Table 4.25: Error Metrics of the Estimated States in the Presence of 11 PMUs

NN State Estimator for unchanged line parameters with Gaussian errors	**NN State Estimator for a change in line parameters with Gaussian errors**	**NN State Estimator for a change in line parameters with GMM and Gaussian errors**
Voltage Magnitude MAE: 0.0013 pu	Voltage Magnitude MAE: 0.0037 pu	Voltage Magnitude MAE: 0.0031 pu

Voltage Angle MAE: 0.2683 degrees Voltage RMSE: 0.0057	Voltage Angle MAE: 0.3028 degrees Voltage RMSE: 0.0066	Voltage Angle MAE: 0.3060 degrees Voltage RMSE: 0.0064

It can be observed that the error metrics increase with the changed line parameters; however, the estimation in the worst case is still relatively accurate. Therefore, it can be concluded that the neural network state estimator is (1) not impacted by the characteristics of the error distribution, and (2) is robust against dynamically varying line parameters.

4.4: State estimation for a 24-hour period

The formulation of a state estimator using neural networks during a particular hour interval was discussed in the previous chapters. The performance of the methodology was demonstrated for the hour interval 07:00 AM-08:00 AM. Now, the states of the system can be estimated for the entire day by formulating a separate neural network for each hour.

The real and reactive load patterns are trained offline for each hour using hourly data obtained for that hour throughout the year. The discrete load points are sampled and the data for neural network is generated using MATPOWER ACOPF as explained in Chapter 3. This process is repeated, and a new neural network is formulated for each hour.

The measurements for the neural network are obtained from the true values by adding errors from the 3 component GMM and Gaussian errors as explained in Chapter 4.2. The neural network is trained, validated, and tested for the IEEE 118-bus system using the measurements from PMUs placed at high voltage buses.

The plot with the voltage magnitude MAE and voltage angle MAE obtained during the different hour intervals are shown in Fig 4.26 and Fig. 4.27 below.

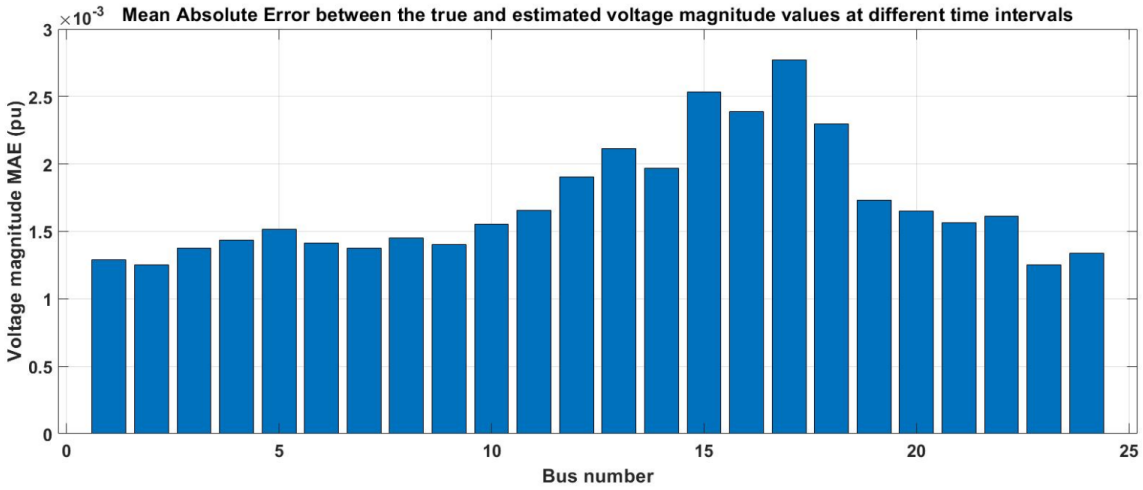

Figure 4.26: Mean Absolute Error between the True and Estimated Voltage Magnitude Values at Different Time Intervals

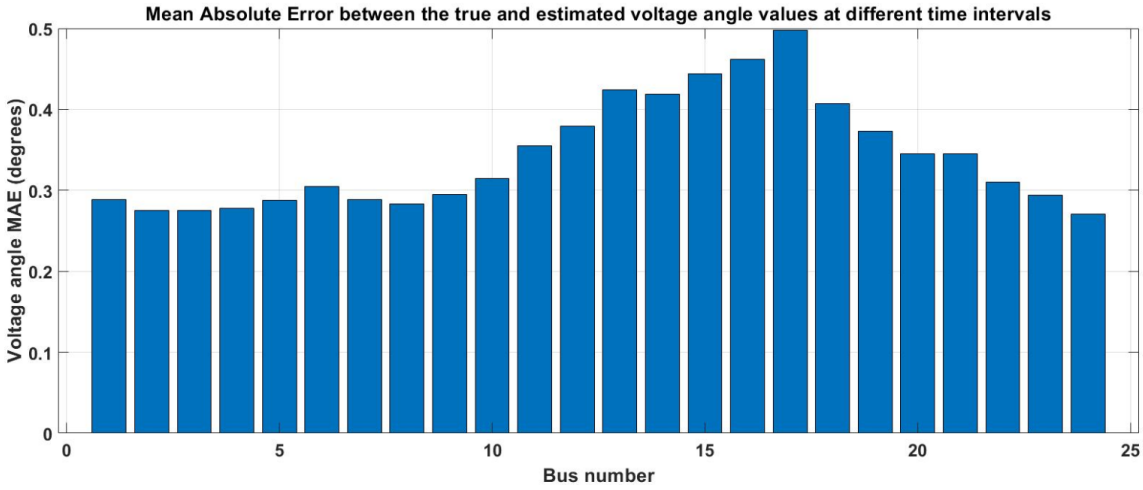

Figure 4.27: Mean Absolute Error between the True and Estimated Voltage Angle Values at Different Time Intervals

It can be realized from Figs. 4.26 and 4.27 that a neural network formulated and trained for each hour can perform accurate state estimation for that corresponding hour over the period of a day even when the system is partially observed for the entire day.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1: Conclusions

An accurate monitoring of the power system is an indispensable requirement for maintaining the reliability of the system. A PMU provides a high-speed, time synchronized complex voltage and current measurements of the buses at which they are placed and the branches connected to these buses. However, the infrastructure required for obtaining the measurements from these devices is expensive. The conventional state estimation techniques that have been proposed previously, which only use PMU measurements can provide high speed snapshots of the power system, only if the system is completely observed by PMUs. A machine learning based formulation using deep neural networks is proposed in this thesis that identifies the joint probability distribution between any number of measurements and the corresponding states using the historical data in an offline process. The highlight of the proposed technique is its ability to estimate the system states at high speeds using real-time synchronized measurements even when the system is not topologically observed by PMUs.

The neural network-based state estimator requires a set of known features and the corresponding labels during the offline training and validation. A load pattern using the hourly load data was obtained using a Kernel Density Estimator and was verified using parametric and non-parametric goodness of fit tests. A procedure to obtain the data for the neural network using load patterns and the formulation of the neural network with different values of hyperparameters for a redundantly observed system was initially discussed. The

state estimation was then performed in the presence of complete observability and partial observability. A MAE of 0.0013 pu for magnitude and 0.2683 degrees for angle was obtained for a partially observable system with PMUs at only the highest voltage buses of the test system (IEEE 118-bus system). The performance of the proposed technique in the presence of errors from GMM and Gaussian distribution, and Laplacian distributions was compared with a classical linear state estimator. It was shown that the proposed estimator performs better even for a completely observed system and additionally it is accurate too for a partially observed system. The robustness of the proposed technique against errors from Non-Gaussian distributions and varying line parameters was also demonstrated in this work. Lastly, the formulation and performance of the state estimator for an entire day was also demonstrated.

5.2: Future Scope of Work

The following are some of the future scopes of work identified for the research done in this thesis:

- The performance of the state estimator with errors in measurements from different distributions was discussed in this work. However, the performance of the state estimator in the presence of bad measurements/missing measurements was not discussed. One future scope of work will be the identification of bad data in the measurements and a methodology to replace them. The statistical techniques used in the identification of bad data in conventional estimators can be applied here too. If the real time PMU measurements are available, a Three-Sample Quadratic Prediction Algorithm [100] can also be used for identifying and replacing bad data.

- The current work discusses the formulation of an estimator under an assumption that the system topology is known and remains unchanged. A future scope of work might consider the formulation and working of a neural network state estimator during a topology change in the system. A deep learning technique called Transfer Learning can be used to handle topology changes.
- The present work requires the formulation of a different state estimator for each hour of the day. A future scope of work might involve reducing the number of state estimators by modeling a graph convolutional neural network with long-short term memory that can learn and predict the sequence of data while accounting for the temporal and spatial attributes of the system.

REFERENCES

[1] California rolling blackouts. [Online]. Available: https://www.politico.com/states/california/story/2020/08/18/california-has-first-rolling-blackouts-in-19-years-and-everyone-faces-blame-1309757

[2] G. S. Vassell, "Northeast Blackout of 1965," *IEEE Power Eng. Review*, vol. 11, no. 1, pp. 4, Jan 1991.

[3] F. C. Schweppe and J. Wildes, "Power System Static-State Estimation, Part I: Exact Model," *IEEE Trans. Power App. Syst.*, vol. PAS-89, no. 1, pp. 120-125, Jan. 1970.

[4] M.B.D.C. Filho, J.C.S.D. Souza, J.D. Glover (2018 Oct. 31), Roots, achievements, and prospects of power system state estimation: A review on handling corrupted measurements. *Wiley*. [Online]. Available: https://onlinelibrary.wiley.com/doi/epdf/10.1002/etep.2779

[5] M. Khorsand, "Dispatch", *Presentation, Power System Operations and Planning*, Tempe, AZ, 2018.

[6] J. Chen, "From State Estimator to SCED", *Presentation*, *ERCOT*, Austin, TX, 12 Nov 2018.

[7] A. Pal, "Instrument Transformers", *Presentation, WAMS in Power System Applications*, Tempe, AZ, 2019.

[8] A. G. Phadke, J. S. Thorp and M. G. Adamiak, "A New Measurement Technique for Tracking Voltage Phasors, Local System Frequency, and Rate of Change of Frequency," *IEEE Trans. Power App. Systems*, vol. PAS-102, no. 5, pp. 1025-1038, May 1983.

[9] A. G. Phadke and J. S. Thorp, Synchronized Phasor Measurements and Their Applications. New York, NY, USA: Springer, 2008, pp. 29–48.

[10] A. Pal, P. Chatterjee, J. S. Thorp, and V. A. Centeno, "On-line calibration of voltage transformers using synchrophasor measurements," *IEEE Trans. Power Del.*, vol. 31, no. 1, pp. 370-380, Feb. 2016.

[11] J. Zheng and A. Dagnino, "An initial study of predictive machine learning analytics on large volumes of historical data for power system applications," *in Proc. IEEE Int. Conf. Big Data*, Washington, DC, 2014, pp. 952-959.

[12] J. Moradi, H. Shahinzadeh, H. Nafisi, M. Marzband and G. B. Gharehpetian, "Attributes of Big Data Analytics for Data-Driven Decision Making in Cyber-

Physical Power Systems," *in Proc. 14th Int. Conf. Protection and Automation of Power Systems (IPAPS)*, Tehran, Iran, 2019, pp. 83-92.

[13] R. Sodhi, S. C. Srivastava, S. N. Singh, "Optimal PMU placement method for complete topological and numerical observability of power system," *Electric Power Syst. Research*, vol. 80, pp. 1154-1159, Apr. 2010.

[14] M. Zhou, V. A. Centeno, J. S. Thorp and A. G. Phadke, "An Alternative for Including Phasor Measurements in State Estimators," *IEEE Trans. Power Syst.*, vol. 21, no. 4, pp. 1930-1937, Nov. 2006.

[15] R. F. Nuqui and A. G. Phadke, "Hybrid Linear State Estimation Utilizing Synchronized Phasor Measurements," *in Proc. IEEE Lausanne Power Tech*, Lausanne, Switzerland, 2007, pp. 1665-1669.

[16] L. Zhang, A. Bose, A. Jampala, V. Madani and J. Giri, "Design, Testing, and Implementation of a Linear State Estimator in a Real Power System," *IEEE Trans. Smart Grid*, vol. 8, no. 4, pp. 1782-1789, Jul 2017.

[17] B. Xu, Y. J. Yoon and A. Abur, "Optimal Placement and Utilization of Phasor Measurements for State Estimation", *in Proc. Power Syst. Computation Conf.*, Liege, Belgium, 2005.

[18] U.S. Department of Energy, Office of Electricity Delivery and Energy Reliability (Oct 2014), 'Factors affecting PMU installation costs'. [Online]. Available: https://www.smartgrid.gov/files/documents/PMU-cost-study-final-10162014.pdf.

[19] A. Pal, A. K. S. Vullikanti and S. S. Ravi, "A PMU Placement Scheme Considering Realistic Costs and Modern Trends in Relaying," *IEEE Trans. on Power Systems*, vol. 32, no. 1, pp. 552-561, Jan. 2017.

[20] M. Ghamsari-Yazdel, M. Esmaili, F. Aminifar, P. Gupta, A. Pal, and H. A. Shayanfar, "Incorporation of controlled islanding scenarios and complex substations in optimal WAMS design," *IEEE Trans. Power Syst.*, vol. 34, no. 5, pp. 3408-3416, Sep. 2019.

[21] C. Mishra, K. D. Jones, A. Pal, and V. A. Centeno, "Binary particle swarm optimisation-based optimal substation coverage algorithm for phasor measurement unit installations in practical systems," *IET Gener. Transm. Distrib.*, vol. 10, no. 2, pp. 555-562, Feb. 2016.

[22] A. Pal, C. Mishra, A. K. S. Vullikanti and S. S. Ravi, "General optimal substation coverage algorithm for phasor measurement unit placement in practical systems," *IET Generation, Transmission & Distribution*, vol. 11, no. 2, pp. 347-353, Jan 2017.

[23] S. Wang, J. Zhao, Z. Huang and R. Diao, "Assessing Gaussian Assumption of PMU Measurement Error Using Field Data," *IEEE Trans. Power Del.*, vol. 33, no. 6, pp. 3233-3236, Dec. 2018.

[24] F. C. Schweppe, "Power System Static-State Estimation, Part III: Implementation," *IEEE Transactions on Power App. Syst.*, vol. PAS-89, no. 1, pp. 130-135, Jan. 1970.

[25] A. Abur and A. Gomez-Exposito, Power System State Estimation: Theory and Implementation. New York: Marcel Dekker, 2004.

[26] G. Hug and J. A. Giampapa, "Vulnerability Assessment of AC State Estimation With Respect to False Data Injection Cyber-Attacks," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1362-1370, Sept. 2012.

[27] M. Meriem, C. Bouchra, B. Abdelaziz, S. O. B. Jamal, E. M. Faissal and C. Nazha, "Study of state estimation using weighted-least-squares method (WLS)," *in Proc. Int. Conf. Electrical Sciences Technologies Maghreb (CISTEM)*, Marrakech, Morocco 2016, pp. 1-5.

[28] A. Monticelli and A. Garcia, "Fast decoupled state estimators," *IEEE Trans. Power Syst.*, vol. 5, no. 2, pp. 556-564, May 1990.

[29] A. Gomez-Exposito, A. Abur, A.D.L.V. Jaen and C. Gomez-Quiles, "A Multilevel State Estimation Paradigm for Smart Grids," *Proc. IEEE*, vol. 99, no. 6, pp. 952-976, June 2011.

[30] G. N. Korres, "A Distributed Multiarea State Estimation," *IEEE Trans. Power Syst.*, vol. 26, no. 1, pp. 73-84, Feb. 2011.

[31] C. Muscas, M. Pau, P. A. Pegoraro, S. Sulis, F. Ponci and A. Monti, "Multiarea Distribution System State Estimation," *IEEE Trans. Instrum. Meas.*, vol. 64, no. 5, pp. 1140-1148, May 2015.

[32] S. C. Müller, A. Kubis, S. Brato, U. Häger, C. Rehtanz and J. Götze, "New applications for wide-area monitoring, protection and control," *in Proc. 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, Berlin, Germany, 2012, pp. 1-8.

[33] M. Li, A. Pal, A. G. Phadke, and J. S. Thorp, “Transient stability prediction based on apparent impedance trajectory recorded by PMUs,” *Int. J. Elect. Power Energy Syst.*, vol. 54, pp. 498-504, Jan. 2014.

[34] A. Pal, J. S. Thorp, T. Khan, and S. S. Young, “Classification trees for complex synchrophasor data,” *Elect. Power Compon. Syst.*, vol. 41, no. 14, pp. 1381-1396, Sep. 2013.

[35] P. Gupta, A. Pal, and V. Vittal, “Coordinated wide-area control of multiple controllers in a power system embedded with HVDC lines,” accepted for publication in *IEEE Transactions on Power Systems*.

[36] M. Padhee, R. S. Biswas, A. Pal, K. Basu, and A. Sen, “Identifying unique power system signatures for determining vulnerability of critical power system assets,” *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 47, no. 4, pp. 8-11, Apr. 2020.

[37] T. Wang, A. Pal, J. S. Thorp, and Y. Yang, “Use of polytopic convexity in developing an adaptive inter-area oscillation damping scheme,” *IEEE Trans. Power Syst.*, vol. 32, no. 4, pp. 2509-2520, Jul. 2017.

[38] T. Wang, A. Pal, J. S. Thorp, Z. Wang, J. Liu, and Y. Yang, “Multi-polytope based adaptive robust damping control in power systems using CART,” *IEEE Trans. Power Syst.*, vol. 30, no. 4, pp. 2063-2072, Jul. 2015.

[39] F. Gao, J. S. Thorp, S. Gao, A. Pal, and K. A. Vance, “A voltage phasor based fault classification method for phasor measurement unit only state estimator output,” *Elect. Power Compon. Syst.*, vol. 43, no. 1, pp. 22-31, Jan. 2015.

[40] M. Barkakati, R. S. Biswas, and A. Pal, “A PMU based islanding detection scheme immune to additive instrumentation channel errors,” in *Proc. IEEE North American Power Symposium (NAPS)*, Wichita, KS, pp. 1-6, 13-15 Oct. 2019.

[41] A. Nath, R. S. Biswas, and A. Pal, “Application of machine learning for online dynamic security assessment in presence of system variability and additive instrumentation errors,” in *Proc. IEEE North American Power Symposium (NAPS)*, Wichita, KS, pp. 1-6, 13-15 Oct. 2019.

[42] P. Gupta, A. Pal, C. Mishra, and T. Wang, “Design of a coordinated wide area damping controller by employing partial state feedback,” in *Proc. IEEE Power Eng. Soc. General Meeting*, Atlanta, GA, pp. 1-5, 4-8 Aug. 2019.

[43] K. Basu, M. Padhee, S. Roy, A. Pal, A. Sen, M. Rhodes, and B. Keel, “Health monitoring of critical power system equipments using identifying codes,” in *Proc.*

13th Int. Conf. Critical Information Infrastructures Security (CRITIS), Kaunas, Lithuania, pp. 1-12, 24-26 Sep. 2018.

[44] V. Chakati, M. Pore, A. Banerjee, A. Pal, and S. K. S. Gupta, "Impact of false data detection on cloud hosted linear state estimator performance," in *Proc. IEEE Power Eng. Soc. General Meeting*, Portland, OR, pp. 1-5, 5-10 Aug. 2018.

[45] V. Chakati, M. Pore, A. Pal, A. Banerjee, and S. K. S. Gupta, "Challenges and trade-offs of a cloud hosted phasor measurement unit-based linear state estimator," in *Proc. IEEE Power Eng. Soc. Conf. Innovative Smart Grid Technol.*, Washington DC, pp. 1-5, 23-26 Apr. 2017.

[46] R. S. Biswas, and A. Pal, "A robust techno-economic analysis of PMU-based islanding detection schemes," in *Proc. IEEE Texas Power Energy Conf. (TPEC)*, College Station, TX, pp. 1-6, 9-10 Feb. 2017.

[47] K. Amare, V. A. Centeno, and A. Pal, "Unified PMU placement algorithm for power systems," in *Proc. IEEE North American Power Symposium (NAPS)*, Charlotte, NC, pp. 1-6, 4-6 Oct. 2015.

[48] C. Mishra, A. Pal, and V. A. Centeno, "Kalman-filter based recursive regression for three-phase line parameter estimation using phasor measurements," in *Proc. IEEE Power Eng. Soc. General Meeting*, Denver, CO, pp. 1-5, 26-30 Jul. 2015.

[49] A. Pal, "Effect of different load models on the three-sample based quadratic prediction algorithm," in *Proc. IEEE Power Eng. Soc. Conf. Innovative Smart Grid Technol.*, Washington D.C, pp. 1-5, 18-20 Feb. 2015.

[50] P. Chatterjee, A. Pal, J. S. Thorp, and J. De La Ree, "Partitioned linear state estimation," in *Proc. IEEE Power Eng. Soc. Conf. Innovative Smart Grid Technol.*, Washington D.C, pp. 1-5, 18-20 Feb. 2015.

[51] A. Pal, I. Singh, and B. Bhargava, "Stress assessment in power systems and its visualization using synchrophasor based metrics," in *Proc. IEEE 2014 North American Power Symposium (NAPS)*, Pullman, WA, pp. 1-6, 7-9 Sep. 2014.

[52] K. Vance, A. Pal, and J. S. Thorp, "A robust control technique for damping inter-area oscillations," in *Proc. IEEE Power Energy Conf. Illinois (PECI)*, Champaign, IL, pp. 1-8, 24-25 Feb. 2012.

[53] F. Gao, J. S. Thorp, A. Pal, and S. Gao, “Dynamic state prediction based on Auto-Regressive (AR) model using PMU data,” in *Proc. IEEE Power Energy Conf. Illinois (PECI)*, Champaign, IL, pp. 1-5, 24-25 Feb. 2012.

[54] A. Pal, and J. S. Thorp, “Co-ordinated control of inter-area oscillations using SMA and LMI,” in *Proc. IEEE Power Eng. Soc. Conf. Innovative Smart Grid Technol.*, Washington D.C., pp. 1-6, 16-20 Jan. 2012.

[55] G. A. Sanchez, A. Pal, V. A. Centeno, and W. C. Flores, “PMU placement for the Central American power network and its possible impacts,” in *Proc. IEEE Power Eng. Soc. Conf. Innovative Smart Grid Technol.*, Medellin, Colombia, pp. 1-7, Oct. 2011.

[56] A. Pal, J. S. Thorp, S. S. Veda, and V. A. Centeno, “Applying a robust control technique to damp low frequency oscillations in the WECC,” *Int. J. Elect. Power Energy Syst.*, vol. 44, no. 1, pp. 638-645, Jan. 2013.

[57] J.D.L. Ree, V. Centeno, J. S. Thorp and A. G. Phadke, "Synchronized Phasor Measurement Applications in Power Systems," *IEEE Trans. Smart Grid*, vol. 1, no. 1, pp. 20-27, Jun 2010.

[58] A. G. Phadke, J. S. Thorp and K. J. Karimi, "State Estimation with Phasor Measurements," *IEEE Trans. Power Syst.*, vol. 1, no. 1, pp. 233-238, Feb. 1986.

[59] S. Roy, H. Chandrasekaran, A. Pal and A. Sen, "A New Model to Analyze Power and Communication System Intra-and-Inter Dependencies," *in Proc. IEEE Conf. Technologies Sustainability (SusTech)*, Santa Ana, CA, 2020, pp. 1-8.

[60] R. Baltensperger, A. Loosli, H. Sauvain, M. Zima, G. Andersson and R. Nuqui, "An implementation of two-stage hybrid state estimation with limited number of PMU," *in Proc. 10th IET Int. Conf. Developments Power Syst. Protection (DPSP 2010)*, Manchester, NH, 2010, pp. 1-5.

[61] A.S. Costa, A. Albuquerque and D. Bez, "An estimation fusion method for including phasor measurements into power system real-time modeling," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 1910-1920, May 2013.

[62] L. Zhao and A. Abur, "Multi area state estimation using synchronized phasor measurements," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 611-617, May 2005.

[63] J. Zhao, G. Zhang and M.L. Scala, "PMU based Robust Dynamic State Estimation method for power systems," *in proc. IEEE Power Energy Soc. General Meeting*, Denver, CO, 2015, pp. 1-5.

[64] J. Zhao, G. Zhang, K. Das, G.N. Korres, N.M. Manousakis, A.K. Sinha, Z. He ., "Power System Real-Time Monitoring by Using PMU-Based Robust State Estimation Method," *IEEE Trans. Smart Grid*, vol. 7, no. 1, pp. 300-309, Jan. 2016.

[65] K. D. Jones, J. S. Thorp, and R. M. Gardner, "Three-phase linear state estimation using phasor measurements," in *Proc. IEEE Power Energy. Soc. General Meeting*, Vancouver, BC, Canada, 2013, pp. 1-5, 21-25.

[66] J. Thorp, "Linear state estimator and synchrophasor data conditioning and validation," *Presentation, WECC Joint Synchronized Information Subcommittee Meeting*, Tempe, AZ, 17 Jan. 2013.

[67] Y. Weng, R. Negi, C. Faloutsos and M. D. Ilić, "Robust Data-Driven State Estimation for Smart Grid," *IEEE Trans. Smart Grid*, vol. 8, no. 4, pp. 1956-1967, Jul 2017.

[68] Y. Weng, R. Negi and M. D. Ilić, "Historical data-driven state estimation for electric power systems," *IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Vancouver, Canada, 2013, pp. 97-102.

[69] L. Zhang, G. Wang and G. B. Giannakis, "Real-Time Power System State Estimation and Forecasting via Deep Unrolled Neural Networks," *IEEE Trans. Signal Process.*, vol. 67, no. 15, pp. 4069-4077, Aug 2019.

[70] L. Zhang, G. Wang and G. B. Giannakis, "REAL-TIME POWER SYSTEM STATE ESTIMATION VIA DEEP UNROLLED NEURAL NETWORKS," *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Anaheim, CA, 2018, pp. 907-911.

[71] A. Abur and M. K. Celik, "A fast algorithm for the weighted least-absolutevalue state estimation (for power systems)," IEEE Trans. Power Syst., vol. 6, no. 1, pp. 1–8, Feb. 1991.

[72] K. R. Mestav, J. Luengo-Rozas and L. Tong, "Bayesian State Estimation for Unobservable Distribution Systems via Deep Learning," *IEEE Trans. Power Syst.*, vol. 34, no. 6, pp. 4910-4920, Nov. 2019.

[73] K. R. Mestav and L. Tong, "Learning the Unobservable: High-Resolution State Estimation via Deep Learning," *57th Annu. Allerton Conf. Commun., Control, Computing (Allerton)*, Monticello, IL, 2019, pp. 171-176.

[74] A. Pal, G. A. Sanchez-Ayala, J. S. Thorp, and V. A. Centeno, "A community-based partitioning approach for PMU placement in large systems," Electr. Power Compon. Syst., vol. 44, no. 12, pp. 1317-1329, Jun. 2016.

[75] Probability Distribution Function. [Online]. Available: https://en.wikipedia.org/wiki/Probability_density_function

[76] 2000 bus synthetic Texas system load data. [Online]. Available: https://electricgrids.engr.tamu.edu/electric-grid-test-cases/activsg2000/

[77] Normal distribution probability density function. [Online]. Available: https://en.wikipedia.org/wiki/Normal_distribution

[78] Weibull distribution probability density function. [Online]. Available: https://en.wikipedia.org/wiki/Weibull_distribution

[79] Log-normal distribution probability density function. [Online]. Available: https://en.wikipedia.org/wiki/Log-normal_distribution

[80] Kernel density estimation probability density function. [Online]. Available: https://en.wikipedia.org/wiki/Kernel_density_estimation

[81] Parametric goodness of fit tests. [Online]. Available: https://statisticsbyjim.com/hypothesis-testing/goodness-fit-tests-discrete-distributions/

[82] L. Wasserman, All of Statistics. Springer-Verlag New York, 2004.

[83] MATPOWER manual. [Online]. Available: https://matpower.org/docs/manual.pdf

[84] C. M. Pacis, G. R. Maramba and C. E. Calamayan, "Experimental design and performance for medium transmission lines using TQ NE9080 line simulator," *in Proc. IEEE Region 10 Conf.*, Penang, Malaysia, 2017, pp. 2347-2352.

[85] "IEEE Standard for Synchrophasor Measurements for Power Systems -- Amendment 1: Modification of Selected Performance Requirements," *IEEE Std C37.118.1a-2014 (Amendment to IEEE Std C37.118.1-2011)* , vol., no., pp.1-25, 30 April 2014.

[86] Number of nodes in hidden layer. [Online]. Available: https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/

[87] D.P. Kingma, J.Ba, " Adam: A new method for stochastic optimization," in Proc. 3rd *Int. Conf. Learning Representations (ICLR),* San Diego, CA, 2015, arXiv:1412.6980.

[88] Keras documentation. [Online]. Available: https://keras.io/

[89] Tensorflow. [Online]. Available: https://www.tensorflow.org/

[90] Pandas python package. [Online]. Available: https://pandas.pydata.org/

[91] B. Xu, A. Abur, “Observability analysis and measurement placement for system with PMUs”, IEEE PES Power Syst. Conf. and Exposition. 2. 943 - 946 vol.2. 10.1109, Nov. 2004.

[92] PMU deployment in power grid. [Online]. Available: https://www.naspi.org/sites/default/files/reference_documents/33.pdf?fileID=1326.

[93] A. Pal, G. A. Sanchez-Ayala, V. A. Centeno and J. S. Thorp, "A PMU Placement Scheme Ensuring Real-Time Monitoring of Critical Buses of the Network," *IEEE Trans. Power Del.*, vol. 29, no. 2, pp. 510-517, Apr 2014.

[94] "IEEE Standard Requirements for Instrument Transformers," *IEEE Std C57.13-2016 (Revision of IEEE Std C57.13-2008)*, vol., no., pp.1-96, Jun 2016.

[95] P. Chatterjee, A. Pal, J. S. Thorp, J. D. L .R. Lopez and V. A. Centeno, "Error reduction of phasor measurement unit data considering practical constraints," *IET Generation Transmission Distribution*, vol. 12, no. 10, pp. 2332-2339, May 2018.

[96] S. Wang, J. Zhao, Z. Huang and R. Diao, "Assessing Gaussian Assumption of PMU Measurement Error Using Field Data," *IEEE Trans. Power Del.*, vol. 33, no. 6, pp. 3233-3236, Dec. 2018.

[97] J. Zhao and L. Mili, "A Framework for Robust Hybrid State Estimation with Unknown Measurement Noise Statistics," IEEE Trans. Industrial Informatics, vol. 14, no. 5, pp. 1866-1875, May 2018.

[98] D. Ritzmann, P. S. Wright, W. Holderbaum and B. Potter, "A Method for Accurate Transmission Line Impedance Parameter Estimation," IEEE Trans. Instrum. Meas., vol. 65, no. 10, pp. 2204-2213, Oct. 2016.

[99] P. K. Mansani, A. Pal, M. Rhodes and B. Keel, "Estimation of Transmission Line Sequence Impedances using Real PMU Data," *in Proc. North American Power Symp. (NAPS)*, Fargo, ND, 2018, pp. 1-6.

[100] K. D. Jones, A. Pal and J. S. Thorp, "Methodology for Performing Synchrophasor Data Conditioning and Validation," *IEEE Trans. Power Syst.*, vol. 30, no. 3, pp. 1121-1130, May 2015.

APPENDIX A

PMU PLACEMENT FORMULATION

The PMU placement formulation for an n-bus system can be formulated as:

$$\min \sum_{i=1}^{n} c_i x_i$$

$$s.t.\ (A * X)\ \geq 1$$

where c_i denotes the cost of installation of PMU at bus i. Also,

$$x_i = \begin{cases} 1 & if\ the\ PMU\ is\ placed\ at\ bus\ i \\ 0 & if\ the\ PMU\ is\ not\ placed\ at\ bus\ i \end{cases}$$

Let F(X) = A*X. The non-zero element of F(X) indicates that the corresponding bus is able to be observed using the optimally placed PMU.

Matrix A is the binary connectivity matrix defined by:

$$A_{i,j} = \begin{cases} 1, & if\ i = j\ or\ if\ i\ and\ j\ are\ connected \\ 0, & otherwise \end{cases}$$

If the cost of the individual PMUs to be placed are all equal, the objective function would be equal to minimizing the total number of PMUs to be placed in the system, to achieve complete system observability.

The above Integer Linear Programming (ILP) formulation can be solved using any available optimization solver. Solving the optimization can lead to multiple set of PMU locations which can lead to minimum PMU placement. In that case, the solution which provides maximum observability over others would be the preferred solution. This can be achieved by solving another optimization problem.

The bus observability index (BOI) β_i is defined as the number of PMUs in an optimally placed solution that can observe a particular bus.

If f(X) = A*X, then

$$\beta_i = f(i)$$

Now, we define SORI (γ) to be sum of observability for all the buses in the system. SORI is mathematically defined as:

$$\gamma = \sum_{i=1}^{n} (\beta_i)$$

Let, k be the number of non-zero values of vector X (optimal number of PMUs) obtained from the above optimization problem. The problem can be formulated as:

$$\text{maximize } \gamma$$

$$s.t.\ (A * X)\ \geq 1$$

$$\sum_{i=1}^{n} (x_i) = k$$

APPENDIX B

NEURAL NETWORK FORMULATION

```
from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Activation

from tensorflow.keras.callbacks import EarlyStopping

from tensorflow.keras.callbacks import ModelCheckpoint

import pandas as pd

import numpy as np

import h5py

from sklearn import metrics

from matplotlib import pyplot as plt

from sklearn.metrics import mean_absolute_error

from sklearn.preprocessing import normalize

from tensorflow.keras.layers import BatchNormalization

from tensorflow.keras.layers import Dropout

from tensorflow.keras.models import load_model

from tensorflow.keras import regularizers

import tensorflow as tf

from sklearn import preprocessing
```

```
# Create the Scaler object

scaler = preprocessing.StandardScaler()

import numpy as np

STR = np.zeros(119)

TL=np.zeros(79)

VL=np.zeros(79)

x=0

num=1

initializer = tf.keras.initializers.he_uniform(seed=None)



X_Train = pd.read_csv(r'D:\Sem 4\New folder\T_Train8IPONN118_fm118.csv')

Y_Train = pd.read_csv(r'D:\Sem 4\New folder\T_Train8IPONN118_o118.csv')

y_train = Y_Train.to_numpy()



x_train = X_Train.to_numpy()



df = pd.DataFrame(x_train)

df.to_csv (r'D:\Sem 4\New folder\T_dummy5file.csv', index = False, header=True)
```

```
X_Val1 = pd.read_csv(r'D:\Sem 4\New folder\T_Val8IPONN118_fm118.csv')

Y_Val = pd.read_csv(r'D:\Sem 4\New folder\T_Val8IPONN118_o118.csv')

y_val = Y_Val.to_numpy()


X_Val = X_Val1.to_numpy()

x_val=X_Val


X_Test1 = pd.read_csv(r'D:\Sem 4\New folder\T_Test8IPONN118_fm118.csv')

Y_Test = pd.read_csv(r'D:\Sem 4\New folder\T_Test8IPONN118_o118.csv')

y_test = Y_Test.to_numpy()


X_Test = X_Test1.to_numpy()

x_test=X_Test



print(x_train.shape)

print(y_train.shape)

print(x_val.shape)
```

```
print(y_val.shape)

print(x_test.shape)

print(y_test.shape)



# Build the neural network

model = Sequential()

#model.add(BatchNormalization())

#model.add(Dropout(0.2))

model.add(Dense(890, kernel_initializer=initializer, input_dim=x_train.shape[1],
activation='relu')) #Hidden 1

model.add(Dense(890, kernel_initializer=initializer, activation='relu')) # Hidden 2

model.add(Dense(890, kernel_initializer=initializer, activation='relu')) # Hidden 3

model.add(Dense(890, kernel_initializer=initializer, activation='relu')) # Hidden 4

model.add(Dense(890, kernel_initializer=initializer, activation='relu')) # Hidden 5



model.add(Dense(236, activation='linear')) # Output

model.compile(loss='mean_squared_error', optimizer='adam')

es = EarlyStopping(monitor='val_loss', patience=50, verbose=1, mode='auto')
```

```
mc = ModelCheckpoint('T_best_model8G118.h5', monitor='val_loss', mode='min', verbose=1, save_best_only=True)

history = model.fit(x_train,y_train,epochs=120, batch_size=8, validation_data=(x_val, y_val), callbacks=[es, mc])

saved_model = load_model('T_best_model8G118.h5')


pred = saved_model.predict(x_test)

score = np.sqrt(metrics.mean_squared_error(pred,y_test))

print(num)

STR[num-1]=score

#print(pred)

print('Final score (RMSE):',score)

print(pred)


df = pd.DataFrame(pred)

df.to_csv (r'D:\Sem 4\New folder\StateEstimates\T_Res8118.csv')

# summarize history for loss

plt.plot(history.history['loss'])
```

```
plt.plot(history.history['val_loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train', 'Validation'], loc='upper left')

plt.show()



df = pd.DataFrame(STR)

df.to_csv (r'D:\Sem 4\New folder\StateEstimates\T_RMSE8118.csv', index = False,
header=True)
```