Stealthy False Data Injection Attack Detection in Power Transmission System using Security Analytics

A Dissertation
Presented in Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

with a

Major in Computer Science

in the

College of Graduate Studies

University of Idaho

by

Mohammad Ashrafuzzaman

Major Professor
Frederick T. Sheldon, Ph.D.

Committee Members
Hasan M. Jamil, Ph.D., Robert B. Heckendorn, Ph.D.,
Daniel Conte de Leon, Ph.D., Michael A. Haney, Ph.D.,
Clinton L. Jeffery, Ph.D.

Department Administrator
Terence Soule, Ph.D.

December 2020

Authorization to Submit Dissertation

This dissertation of Mohammad Ashrafuzzaman, submitted for the degree of Doctor of Philosophy with a Major in Computer Science and titled “Stealthy False Data Injection Attack Detection in Power Transmission System using Security Analytics,” has been reviewed in final form. Permission, as indicated by the signatures and dates below, is now granted to submit final copies to the College of Graduate Studies for approval.

Major Professor:	Frederick T. Sheldon, Ph.D.	Date
Committee Members:	Hasan M. Jamil, Ph.D.	Date
	Robert Heckendorn, Ph.D.	Date
	Daniel Conte de Leon, Ph.D.	Date
	Michael A. Haney, Ph.D.	Date
	Clinton L. Jeffery, Ph.D.	Date
Department Administrator:	Terence Soule, Ph.D.	Date

ABSTRACT

The electric smart grid, a critical national infrastructure and among the largest and most complex cyber-physical systems, is under constant and multifarious threat of cyber-attacks. State estimation (SE) is at the foundation of a series of critical control processes in a power transmission system. A sophisticated cyber-attacker can intelligently change the values in the measurement matrix used to compute state estimation. These data integrity attacks can potentially disrupt the critical control processes, adversely affecting a power system operationally and economically. Stealthy false data injection (SFDI) attacks against SE cannot be detected by the conventional bad-data detection mechanisms.

In this dissertation, a security analytics framework to detect SFDI attacks on static SE measurement data is presented. A threat model that identified three possible attack models was developed, and synthetic datasets corresponding to these attack models were generated for standard IEEE 14-bus and 57-bus systems. After normalizing and reducing the number of features in the datasets, a number of supervised, unsupervised, and stacking ensemble machine learning models were trained and tested for model selection. Through the model selection process, including hyper-parameter tuning and cross-validation, trained models were identified that can detect the SFDI attacks accurately and reliably. Evaluation of the models using standard metrics shows that supervised artificial neural networks with four hidden layers and 1200 hidden units per layer can detect 98.24% of the attacks with a false alarm rate of 1.25%. Among the unsupervised models, elliptic envelope performs the best with 73% detection rate with 3% false alarm rate. It was also found that the detection rate is the same for all the machine learning methods for all the six datasets corresponding to different attack models and bus systems.

The core contributions of this dissertation are the demonstration that a machine learning based security analytics framework can successfully detect the SFDIA attacks and the identification of artificial neural network with the right set of hyper-parameter values as the best performing model. Additional contributions include a survey and a taxonomy of false data injection attacks on different parts of the power grid for the first time in the literature, an exhaustive survey of machine learning based approaches for detecting SFDI attacks, implementation of a software for running the machine learning models, and identification of a number of research ideas based on this research.

Acknowledgements

The formulation of this research proposal has been possible because of the support from many sources, and I wholeheartedly acknowledge my indebtedness to all of them. I express my deepest sense of gratitude and appreciation to my major professor Dr. Frederick T. Sheldon for his encouragement, guidance and support in undertaking and proceeding with this research. I extend my heartfelt gratitude towards the members of my dissertation committee, Dr. Clinton Jeffery, Dr. Robert Heckendorn, Dr. Hasan Jamil, Dr. Daniel Conte de Leon, and Dr. Michael Haney for their valuable advice during the research, for reviewing this dissertation and for providing insightful feedback. I am grateful to Dr. Yacine Chakhchoukh for his support in generating the all-important datasets for this research, and to Dr. Brian Johnson for his many important advice. I gratefully acknowledge the research collaborations with Dr. Ananth Jillepalli and Saikat Das and their encouragement. My sincere thanks go to Victor House and John McFarland for providing technical support in setting up the research infrastructure, and to Susan Branting and Arleen Furedy for providing administrative support. This research was partially supported by an Idaho Global Entrepreneurial Mission (IGEM) Grant for Security Management of Cyber-Physical Control Systems, 2016 (Grant Number IGEM17-001). I gratefully acknowledge this support from the State of Idaho.

Dedication

To my parents Dil Afroz Laila and Khandakar Zahurul Islam

and

To my wife Raihan Akhter

and

To my children Wazeeha Ashraf and Zahur Ashrafuzzaman

Table of Contents

LIST OF FIGURES

List of Tables

List of Abbreviations

Table 1: List of abbreviations used in the dissertation.

Abbreviation	Term	Abbreviation	Term
AC	alternating current	IRP	incident response plan
ADMM	alternating direction method of multipliers	IT	information technology
AE	autoencoder	KLD	Kullback-Leibler distance
AGC	automatic generation control	kNN	k-nearest neighbors
AMI	advanced metering infrastructure	KPCA	kernel PCA
ANN	artificial neural network	LMP	locational marginal pricing
BDD	bad data detection	LR	logistic regression
CDBN	conditional deep belief network	LSTM	long short-term memory
CGB	conditional Gaussian-Bernoulli	MGD	mixed Gaussian distribution
CNN	convolutional neural network	ML	machine learning
CPS	cyber-physical system	MLP	multi-layer perceptron
DC	direct current	MSA	margin-setting algorithm
DRE	density ratio estimation	MSE	mean squared error
DRF	distributed random forest	MV	majority voting
DT	decision tree	NAN	neighborhood area network
EBL	explanation-based learning	NB	naïve Bayes
ED	Euclidean distance	NN	neural network
EE	elliptic envelope	OCSVM	one-class SVM
EEN	edited nearest neighbor	OCSVM_L	OCSVM with linear kernel
ELM	extreme learning machine	OCSVM_P	OCSVM with polynomial kernel
EMS	energy management system	OPF	optimal power flow
ENN	extended nearest neighbor	OT	operational technology
Ens_DT	ensemble with decision tree	PCA	principal component analysis
Ens_LR	ensemble with logistic regression	PMU	phasor measurement unit
Ens_MV	ensemble with majority voting	PTS	power transmission system
Ens_NB	ensemble with naïve Bayes	RBM	restricted Boltzmann machine
Ens_NN	ensemble with neural network	RC	robust covariance
Ens_SVM	ensemble with SVM	RFC	random forest classifier
ERT	extremely randomized tree	RLR	robust logistic regression
FAR	false alarm rate (same as FPR)	RNN	recurrent neural network
FCM	fuzzy C-means	ROC	receiver operating characteristic
FDI	false data injection	RTU	remote terminal unit
FDIA	false data injection attack	S3VM	semi-supervised SVM
FN	false negative	SAE	stacked autoencoder
FP	false positive	SARSA	state–action–reward–state–action
FPR	false positive rate	SCED	security-constrained economic dispatch
FRTU	feeder remote terminal unit	SE	state estimation
GA	genetic algorithm	SFDI	stealthy false data injection
GBM	gradient boosting machine	SFDIA	stealthy false data injection attacks
GLM	generalized linear model	SG	smart grid
GPS	global positioning system	SLR	sparse logistic regression
GPU	graphical processing unit	SSA	static security assessment
HAN	home area network	SVM	support vector machine
HPC	high-performance computing	TN	true negative
ISOF	isolation forest	TP	true positive
ICA	independent component analysis	TPR	true positive rate
ICT	information & communication technology	WLS	weighted least square

CHAPTER 1 INTRODUCTION

1.1 THE CONTEXT

The power grid[1], including generators, transmission systems, distribution systems, and numerous other devices, is one of the largest and most complex critical infrastructures. The power grid and its various components have, for decades, been undergoing evolution. Significant changes came when the components of the grid and the supervisory control and data acquisition (SCADA) systems that monitor and control these components were updated with networking capabilities. This essentially transformed power grids into cyber-physical systems (CPS), with the downside of inheriting issues associated with being "cyber" (e.g., vulnerability to exploitation by cyber-attackers). Nonetheless, the industry has attempted to "air-gap" operational technology (OT) from information technology (IT) networks toward protecting valuable CPS assets critical to stable operations. Unfortunately, many OT networks are still not fully insulated from the IT networks [16] and are vulnerable to both internal and external threats [77]. A study by the Ponemon Institute reports that 90% of organizations relying on OT have experienced at least one business-impacting cyber-attack within the 2 years prior to the report [92]. As a result the power grid has been subjected to a new set of exploits, in addition to the ones already present due to the complexity of internetworking of SCADA systems [35, 89, 104, 130].

The vulnerabilities of power grids are illustrated with a few well-known incidents. In January of 2008 the Central Intelligence Agency (CIA) reported that a number of

[1]throughout this dissertation, the terms "smart grid" and "power grid" are used interchangeably.

non-US cities were under cyber-attack affecting distribution systems causing a widespread blackout [1]. In December 2015, three power distribution companies were taken down in a coordinated cyber-attack where seven substations were isolated for 7 hours and operators were forced to switch to manual mode in order to gain control back to the grid, resulting in a power outage for about 225,000 Ukrainians [61].

1.2 THE RESEARCH PROBLEM

One of the many ways smart grids can be attacked over the cyber network is using stealthy false data injection (SFDI) attacks, which are described as a new class of cyber-attacks in power grids by Liu et al. [74]. In SFDI attacks, also called deception attacks or covert data integrity attacks, adversaries intrude into different parts of the power grids, for example smart meters, substation meters or sensors, any part of the cyber-network, i.e., wired or wireless communication channels, routers, SCADA control system computers, etc. and then inject or modify any measurement data with the goal of causing disruption in the grid operation, power theft, or any other malicious intent. As shown by Xiang et al. [116], false data injection is can be an important element of a coordinated attack on the power grid and represents an important class of attack on cyber-physical systems.

One such SFDI attack targets the state estimation process in the power transmission systems. State estimation (SE) is a fundamental tool in the energy management system (EMS) at the power grid control center. The SE computes voltage magnitudes and phase angles at all of the different buses of the power system after collecting measurements that are communicated to the control center from remote terminal units (RTUs) equipped with SCADA units [2]. State estimation process is described in Section 3.2. If the incorrect measurement data affect the outcome of state estimation, the resulting misinformation can reduce the control center operators' level of situational awareness [6]. This potentially forces the operators to take corrective actions under the false assumption that the state variable values obtained from state estimation are correct. This may cause disruption in the real-time operation of the transmission system by adversely impacting tools for contingency analysis, unit commitment, optimal power flow and computation of locational marginal pricing (LMPs) for electricity markets. Cyber-attacks that impact the SE results have been presented in several publications [56, 62, 72, 74].

The state estimation process includes a step called "bad data processing" whereby any anomalous data, be it due to telemetry malfunction or false data injection (FDI) attack, is corrected or suppressed. However, there is a subclass of FDI attacks, termed stealthy false data injection (SFDI) attacks that cannot be detected by the traditional bad data detection mechanisms based on residual analysis. An explanation of what constitutes "stealthy" false data injection attacks on the state estimation in power transmission system is given in Section 3.3. Efficient detection of SFDI attacks on the state estimation is an active research area.

1.3 SECURITY ANALYTICS APPROACH

Rapid advancement in machine learning algorithms have enabled them to find natural patterns in data that generate insight and enable better decisions and predictions. The use of data analytics to predict, detect, and prevent security threats is termed *security analytics* [25]. The incorporation of cyber capabilities into smart grid functionality has led to the proliferation of new data sources. The availability of abundant data generated by these components has enabled investigators to better study cybersecurity threats and countermeasures in smart grids using security analytics [104].

After using traditional statistical approaches and physics of state estimation, researchers have embarked on using machine learning based approaches to detect SFDI attacks. A literature review on machine learning based stealthy false data injection attack (SFDIA) detection approaches given in Chapter 2 shows that this research area is still far from being mature, and a natural progression seems toward using machine learning (ML) based approaches to develop an effective way to detect the SFDI attacks on the state estimation in power transmission systems [85].

In this dissertation research, an empirical-based security analytics approach is used to investigate and identify suitable machine learning methods that will reliably and accurately detect presence of SFDI attacks in the state estimation measurement data. The steps followed in this approach are enumerated below.

1. As part of threat modeling, three attack models for stealthy false data injection attacks on the state estimation in power transmission systems were identified depending on how many RTUs are compromised.

2. Standard IEEE power transmission systems with 14 buses and 57 buses were simulated using MATLAB MATPOWER. This simulation generated power flow measurement data used by state estimation under normal operation of the systems.

3. An attack generation module was used to generate spoofed data that replaced actual data in some of the normal measurement data, simulating stealthy false data injection attacks. For each of the three attack models, datasets were gen-

erated and attack data were injected in the SE measurement data, obtaining six datasets (three per IEEE bus system).

4. The features (i.e., the sensor measurements) in the datasets were ranked according to their importance (in training machine learning models) using random forest classifier and eliminated the features that have minimum contribution on model training.

5. Three supervised machine learning models, namely GLM, GLB, and DRF, were trained, tested, and evaluated for performance using standard metrics to find out the efficacy of these models in detecting SFDI attacks.

6. Two stacking ensembles using supervised and unsupervised machine learning models were developed. The ensemble models were trained, tested, and evaluated for performance using standard metrics to find out the efficacy of these models in detecting SFDI attacks.

7. Unsupervised elliptic envelope method with different hyper-parameter settings was trained, tested and evaluated for performance using standard metrics to find out the efficacy of these models in detecting SFDI attacks.

8. Supervised artificial neural networks with various hyper-parameter settings were trained, tested and evaluated for performance using standard metrics to find out the efficacy of these models in detecting SFDI attacks.

9. After analyzing and comparing the performance of the different models in detecting SFDI attacks, the best-performing models were identified. It was found that artificial neural network performs the best, among supervised models, with 98.25% detection rate and 1.25% false alarm rate and elliptic envelope performs

the best among unsupervised models with 73% detection rate and 3% false alarm rate on an average over all the six datasets.

1.4 SCOPE OF THE RESEARCH

This research is scoped by the following constraints:

1. The research considered SFDIA on only static state estimation in AC power transmission systems.

2. The research considered time-discrete data only, and not time-series data.

3. The research was performed using simulated dataset, because of non-availability of real data.

1.5 CONTRIBUTIONS

The main contributions of the dissertation are:

1. Use of a security analytics based framework to analyze power transmission system state estimation measurement data to detect presence of stealthy false data injection attacks.

2. Using the framework, achieving over 98% detection rates with negligible false alarm rates, which is among the best in the literature.

Other contributions of the dissertation are:

1. An exhaustive survey of machine learning based approaches for detecting SFDI attacks, as presented in Chapter 2.

2. A taxonomy and survey of SFDI attacks on different components of the smart grid, as presented in Chapter 4.

3. Developing a guideline for generating simulated measurement data with SFDI attack, as described in Section 5.5.

4. Implementation of a software tool as a framework for running different machine learning models on a dataset. The framework runs the ML methods with different hyper-parameter settings one by one, collects different evaluation metrics for the methods, and prints out the collated results in tables and as graphs. (Described in Section 5.11).

5. Model selection and evaluation of SFDI attack detection efficiency of three supervised methods, namely GLM, GLB, and DRF, as described in Chapter 6.

6. Use of stacking ensemble models for detecting SFDI attacks, as described in Chapter 7. The ensemble framework consists of two stacking ensembles: one using five supervised methods in the first stage of the stacks, and another using five unsupervised models. Six binary classifiers were used in the second stage as ensemble classifiers.

7. Model selection and evaluation of SFDI attack detection efficiency of artificial neural network with varying hyper-parameter values to find out the best-performing model, as described in Chapter 8.

8. Model selection and evaluation of SFDI attack detection efficiency of elliptic envelope unsupervised method with varying hyper-parameter values to find out the best-performing model, as described in Chapter 9.

9. Identifying a number of research that can be undertaken as direct consequences of this research, as enumerated in Section 10.3.

1.6 AUTHOR'S RELATED PUBLICATIONS

1. M. Ashrafuzzaman, S. Das, Y. Chakhchoukh, S. Shiva, and F. T. Sheldon. "Detecting stealthy false data injection attacks in the smart grid using ensemble-based machine learning," Journal of Computers & Security, Elsevier, August 2020. doi:10.1016/j.cose.2020.101994

2. M. Ashrafuzzaman, S. Das, Y. Chakhchoukh, S. Duraibi, S. Shiva, and F. T. Sheldon. "Supervised learning for detecting stealthy false data injection attacks in the smart grid," Transactions on Computational Science and Computational Intelligence, Advances in Security, Networks, and Internet of Things, Springer Nature, 2020.

3. M. Ashrafuzzaman, S. Das, A. Jillepalli, Y. Chakhchoukh, and F. T. Sheldon. "Elliptic envelope for detecting stealthy false data injection attacks in the smart grid control systems," in IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, December 2020.

4. M. Ashrafuzzaman, Y. Chakhchoukh, A. Jillepalli, P. Tosic, D. Conte de Leon, F. Sheldon, and B. Johnson, "Detecting stealthy false data injection attacks in power grids using deep learning," in IEEE Wireless Communications and Mobile Computing Conference (IWCMC), 14th International, pp. 219–225, IEEE, 2018. doi:10.1109/IWCMC.2018.8450487

5. M. Ashrafuzzaman, H. Jamil, Y. Chakhchoukh, and F.T. Sheldon, "A best-effort damage mitigation model for cyber-attacks on smart grids," 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), vol. 2, pp. 510-515, IEEE 2018. doi:10.1109/COMPSAC.2018.10285

The author's other collaborative publications not directly related to this dissertation are:

1. V. Koganti, M. Ashrafuzzaman, A. Jillepalli and F.T. Sheldon, "A virtual testbed for security management of cyber-physical control systems," IEEE 12th International Malicious and Unwanted Software Conference (MALCON), Puerto Rico, 11-14 October 2017.

2. S. Das, M. Ashrafuzzaman, F. T. Sheldon, and S. Shiva, "Network intrusion detection using natural language processing and ensemble machine learning," in IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, December 2020. (accepted)

3. I. Oyewumi, A. Jillepalli, P. Richardson, M. Ashrafuzzaman, B. Johnson, Y. Chakhchoukh, M. Haney, F. Sheldon, and D. Conte de Leon, "ISAAC: The Idaho CPS smart grid cybersecurity testbed," IEEE Texas Power and Energy Conference (TPEC), 2019.

4. A. Jillepalli, D. Conte de Leon, B. Johnson, Y. Chakhchoukh, I. Oyewumi, M. Ashrafuzzaman, F. Sheldon, J. Alves-Foss and M. Haney, "METICS: A holistic cyber-physical system model for IEEE 14-bus power system security," 13th International Conference on Malicious and Unwanted Software (MALCON), 2018.

5. A. Jillepalli, D. Conte de Leon, M. Ashrafuzzaman, Y. Chakhchoukh, B. Johnson, F. Sheldon, J. Alves-Foss, P. Tosic and M. Haney, “HESTIA: Adversarial modeling and risk assessment for cyber-physical control systems,” 14th IEEE International Wireless Communications and Mobile Computing Conference (IWCMC), Limassol, Cyprus, June 2018.

6. A. Jillepalli, D. Conte de Leon, Y. Chakhchoukh, M. Ashrafuzzaman, B. Johnson, F. Sheldon, J. Alves-Foss, P. Tosic and M. Haney, “An Architecture for HESTIA: High-level and extensible system for training and infrastructure risk assessment,” International Journal of Internet of Things and Cyber Assurance, 2018.

1.7 ORGANIZATION OF THE DISSERTATION

The remainder of the dissertation is organized as follows. Chapter 2 summarizes the related work that used machine learning methods for detecting SFDI attacks on the SE. Chapter 3 provides a description of state estimation in power transmission systems, SFDI attacks on state estimation, and the SFDI attack process. Chapter 4 presents a taxonomy and a survey of SFDI attacks in smart grid. Chapter 5 describes the security analytics based SFDIA detection framework put forward in this dissertation. Chapter 6 describes the use of supervised methods GLM, GLB, and DRF to detect SFDIA and presents the corresponding results. Chapter 7 describes the use of stacking ensemble models to detect SFDIA and presents the corresponding results. Chapter 8 describes the use of artificial neural networks and corresponding results. Chapter 9 describes the use of of unsupervised elliptic envelope method and corresponding results. The dissertation is concluded in Chapter 10, followed by the References. Detail results from the machine learning model selections and evaluations in Chapters 7–9 are presented in Appendix A. An outline of a damage mitigation model is described in Appendix B.

CHAPTER 2 RELATED WORK

2.1 INTRODUCTION

The literature is replete with detection of SFDIA using different machine learning methods, however none has described the complete process using security analytics. Therefore, in this chapter, a review of the literature for machine learning-based approaches to detect stealthy false data injection attacks on the state estimation of power grids is presented. Table 2.1 shows the learning class, algorithms used for feature selections, algorithms used for training, and how the datasets were generated or obtained for each of the works reviewed. The table also lists the metrics used by different works to evaluate their proposed methods.

2.2 DETECTION USING SUPERVISED LEARNING

In supervised learning, labeled data, i.e., a training set of examples with correct responses or ground truths, is provided and based on this training the machine learning algorithm generalizes (i.e., learns the patterns in the training data) respond correctly to input sets that are unlabeled. In this section, the supervised methods used for detecting SFDIA on the state estimation in power transmission systems are described.

The performance comparison of several supervised learning methods by Ozay et al. [88] is one of the early works that used machine learning for detection of SFDIA. They ran multi-layer perceptron (MLP), k-nearest neighbours (kNN), support vector machine (SVM), sparse logistic regression (SLR), and AdaBoost and gave comparative

TABLE 2.1: Literature review of machine learning-based detection of false data injection attacks on the state estimation.

ML Class	Author(s)	Feature . Selections	Training Models	Datasets Generator	Performance Metrics
Supervised	Ozay et al. [88]		MLP, kNN, SVM, SLR, & AdaBoost	MATPOWER	Sensitivity, Precision, & Accuracy
	He et al. [43]	CGB-RBM	CDBN	MATPOWER	Accuracy
	Wang et al. [112]		MSA	Simulink	Accuracy
	Wang et al. [110]		kNN, ANN, SVM, NB, & DT	MATPOWER	Sensitivity, Precision, & Accuracy
	Ashrafuzzaman et al. [11]	RFC	GLM, GBM, DRF & ANN	MATPOWER	Sensitivity, Precision, FPR, & F1-score
	Ayad et al. [12]		RNN	MATPOWER	Sensitivity, Precision, & Specificity
	Ahmed et al. [3]	GA	ED-based	MATPOWER	Accuracy, & F1-score
	Ahmed et al. [4]	GA	SVM	MATPOWER	Accuracy, & F1-score
	Niu et al. [87]		RNN & CNN	MATPOWER	Accuracy
	Wang et al. [108]		RF, AdaBoost	Simulated	Accuracy, Precision, & Sensitivity
	Alimi et al. [7]		SVM+ANN	Nigerian Grid	Sensitivity, Precision, & F1-score
	Yang et al. [122]	AE	ELM	MATPOWER	Sensitivity, Precision, & F1-score
	Camana-Acosta et al. [17]	KPCA	ERT	MATPOWER	Accuracy
	Mohammadpourfard et al. [80]	PCA	kNN	MATPOWER	FPR & F1-score
	Yan et al. [120]		SVM, kNN, & ENN	MATPOWER	Accuracy & F1-score
	Ganjkhani et al. [33]		NARX-ANN	MATPOWER	MSE
	Xu et al. [119]		RNN	MATPOWER	Sensitivity, Precision, FPR, & F1-score
	Hamlich et al. [39]		kNN, RF, DT, MLP, & SVM	Physical Data	Accuracy
Unsupervised	Chaojun et al. [20]		KLD-based	MATPOWER	Sensitivity
	Hao et al. [41]		PCA-approximation	MATPOWER	Accuracy & FPR
	Ahmed et al. [3]	GA	ED-based	MATPOWER	Accuracy & F1-score
	Wang et al. [109]	SAE	LR	MATPOWER	Sensitivity
	Yang et al. [121]	PCA	OCSVM, RC, ISOF, & LOF	MATPOWER	Accuracy & Precision
	Ahmed et al. [5]	PCA	Isolation Forest	MATPOWER	Accuracy & F1-score
	Mohammadpourfard et al. [79]	PCA	Fuzzy C-Means	MATPOWER	Sensitivity
Semi-supervised	Ozay et al. [88]		S3VM & SLR	MATPOWER	Sensitivity, Precision, & Accuracy
	Esmalifalak et al. [29]	PCA	Distributed SVM	MATPOWER	F1-score
	Chakhchoukh et al. [19]		DRE	MATPOWER	Sensitivity
	Foroutan & Salmasi [32]	PCA	MGD-based	MATPOWER	F1-score
Reinforcement	Kurt et al. [57]		SARSA	MATPOWER	Sensitivity, Precision, & F1-score

performances of these models. They used simulated data from MATPOWER [132]. They reported performance of the models as a function of sparsity of attacks. Depending on the sparsity the performance for different models change a lot. They showed that with the least sparse attack model AdaBoost can achieve 100% precision.

He et al. [43] proposed a conditional deep belief network (CDBN), one of various deep neural network architectures, to efficiently reveal the high-dimensional temporal behavior features of the stealthy FDI attacks. The CDBN they employed uses conditional Gaussian-Bernoulli restricted Botzmann machines (CGB-RBM) for the first hidden layer to extract the high-dimensional temporal features. In all other hidden layers conventional RBM were used. They used IEEE 118-bus system simulated using MATPOWER to validate their proposed solution. They showed that the attack detection increases with increase in number of buses. The highest accuracy value of 98.10% is obtained by a CDBN with 5 hidden layers, which is only a 0.5% improvement over a 3-layer CDBN.

Wang et al. [110] applied the concept of “first difference”, borrowed from economics and statistics to time-series measurement data to detect time-synchronous attacks on the measurements. The “first-difference aware” data is then trained using supervised models kNN, artificial neural network (ANN), SVM, naïve Bayes (NB), and decision tree (DT). They tested their proposed approach on MATPOWER-simulated IEEE 14-bus system. They achieved 99.73% overall accuracy with an ANN model.

Ashrafuzzaman et al. [11] proposed a feed-forward neural networks (FFNN) based scheme with different configurations for detecting stealthy FDI attacks on the state estimation of a power grid. They used random forest for feature selection and compared performances of the deep learning scheme with three other machine learning algorithms, namely gradient boosting machines (GBM), generalized linear models

(GLM) and distributed random forests classifier (DRF). Like others they conducted the experiments using data generated for an IEEE 14-bus system using MATPOWER.

Ahmed et al. [3] proposed two Euclidean distance-based anomaly detection schemes. The first scheme utilizes unsupervised-learning over unlabeled data to detect outliers or deviations in the measurements. The second scheme employs supervised-learning over labeled data to detect the deviations in the measurements. The authors used a genetic algorithm for feature selection. They have tested the proposed methods on IEEE 14-, 39-, 57- and 118-bus systems using MATPOWER generated data.

The proposed framework by Niu et al. [87] has two detectors: a network anomaly detector and an FDI attack detector. For detecting the FDI attacks, they formulated these attacks as time-series anomaly and used an LSTM-based convolutional neural network (CNN). At the same time a recurrent neural network with LSTM cell is deployed to capture the dynamic behavior of the cyber-networks in the power system. They tested their model on an IEEE 39-bus system simulated by MATPOWER.

Camana-Acosta et al. [17] proposed a classification scheme based on the extremely randomized trees (ERT) algorithm and kernel principal component analysis (KPCA) for dimensionality reduction. They evaluated the proposed scheme using MATPOWER-simulated standard IEEE 5- and 118-bus systems.

Mohammadpourfard et al. [80] used the idea of *concept drift*, unpredictable shifts in the underlying distribution of historical data over time, to develop a technique to improve the performance of existing supervised learning methods in detecting SFDI attacks. They tested their idea using kNN with PCA as feature extractor and evaluated the performance using a MATPOWER-simulated standard IEEE 14-bus system.

Ganjkhani et al. [33] took into account the high correlation between the power flow measurements data as well as among the state variables and proposed a method that uses a recurrent architecture of ANN configured with nonlinear auto-regressive exogenous (NARX). They simulated an IEEE 14-bus system and used the corresponding data to validate their model.

Hamlich et al. [39] used five classifiers, namely kNN, random forest (RF), decision trees (DT), MLP and SVM, for detecting SFDIA. They collected power flow data from a physical bus feeding system connected to a power system stabilizer and then used MATPOWER to introduce the false measurements.

2.3 DETECTION USING UNSUPERVISED LEARNING

In this section, the unsupervised methods used for detecting SFDIA on the state estimation in power transmission systems are described. Unsupervised methods are trained with unlabeled datasets. Instead of relying on ground truths, these methods attempt to learn the intrinsic properties of the data, and separates the data into regions or clusters of data that have similar properties. In other words, the algorithms attempt to learn the hidden patterns in the input data, and later predicts responses to test inputs based on the learned patterns.

Chaojun et al. [20] proposed a method that tracks the dynamics of variations in the measurement data. They used Kullback-Leibler distance (KLD), that calculates the distance between two probability distributions derived from measurement variations, for tracking that dynamics. The method detects presence of SFDIA when KLD is larger than those for the historical data. They generate the attack data for an IEEE 14-bus system by modifying load data from the New York independent system operator. The method fails to detect SFDIA in system buses where the change in measurement data is very small.

Wang et al [109] developed an interval state estimation based defense mechanism. It's a two step sparse model. First, the lower and upper bounds of each state variable are modeled as a dual optimization problem that aims to maximize the variation intervals of the system variable. Then stacked auto-encoder (SAE), is used to extract the nonlinear and non-stationary features in electric load data. The 6-layer SAE had logistic regression as the final layer. They also used MATPOWER to simulate 9-bus, 14-bus, 30-bus and 118-bus systems, and generated attack data using MATLAB. Their dataset is also quite small, 35,000, for any deep learning training. The training phase takes just over 12 minutes. While this work deals with data in the power systems, the goal of the paper was to predict electric load forecasting more accurately, not to detect SFDI attacks.

Hao et al. [41] proposed a sparse PCA-approximation based model to detect stealthy FDI attacks. In this model, identification of real measurements with the availability of sparse data sets is achieved by using recovery functions. The recovery function's accuracy is inversely proportional to the sparsity of available data. As such, this model falls short at identifying FDI attacks when data is too sparse to produce reliably accurate recovery functions. They evaluated their approach on IEEE 9-, 14-, and 57-bus systems simulated with MATPOWER. They showed that the probability of successful detection of attack increases if the system's error tolerance performance is relaxed. If the system can tolerate a false alarm rate of 10%, then attack detection accuracy approaches 100%.

Ahmed et al. [5] utilized unsupervised learning method isolation forest to detect FDI attacks–they call it covert data integrity assault–using simulated data generated by MATPOWER. They reduce the dimensionality of the data using principal component analysis (PCA). In order to demonstrate that isolation forest performs better, they compare the results with performance of a few other learning models namely SVM, kNN, NB and MLP. They did not report how long it took to train the models. It is unexpected that their results of isolation forest is better than the other models which are all supervised models. That is because supervised models generally perform better in terms of accuracy than unsupervised models on the same dataset. They showed accuracy of up to 94.67% and F1-score of up to 94.41%.

Mohammadpourfard et al. [79] used Fuzzy C-Means (FCM) too detect SFDIA by measuring and comparing deviations in the probability distributions in the data. They used PCA for feature reduction. They generated the attack data for IEEE 14-bus and IEEE 9-bus systems by replacing load data from the New York independent system operator with spoofed data. They reported that their method can achieve 92.88% detection rate.

Ahmed et al. [5] utilized unsupervised learning method isolation forest (ISOF) to detect FDI attacks using simulated data generated by MATPOWER. They reduced the dimensionality of the data using principal component analysis (PCA). To demonstrate that ISOF performs better, they compared their results with those of a few other learning methods namely support vector machines (SVM), k-nearest neighbors (k-NN), naive Bayes (NB) and multilayer perceptron (MLP). They did not report how long it took to train the models. They reported only accuracy, precision and F1-score values. It is unexpected that their results of ISOF are better than the other models which are all supervised models. Generally, supervised models perform better in terms of accuracy and precision than unsupervised models on the same dataset because they are trained with labeled data. In a separate work, Ahmed et al. [3] proposed a Euclidean distance-based anomaly detection scheme. The authors used a genetic algorithm for feature selection. They tested the proposed methods on IEEE 14-, 39-, 57- and 118-bus systems using MATPOWER generated data.

Yang et al. [121] used one-class SVM (OCSVM), robust covariance, ISOF and local outlier factor (LOF) methods. They ran these methods using data from a simulated IEEE 14-bus system. However, the dataset uses only 1000 set of measurements. They reported only accuracy and precision values for the algorithms which can be misleading metrics for anomaly detection. The most relevant metrics for sparse attack detection are sensitivity or recall, specificity, and F1-score.

2.4 DETECTION USING SEMI-SUPERVISED LEARNING

In semi-supervised learning, the models are trained using a small amount of labeled data along with a large amount of unlabeled data during training. These methods are suitable when labeled data is sparse and the training with small amount of labeled data can considerably improve training accuracy. In this section, the works that used semi-supervised methods for detection of SFDIA are summarized.

Esmalifalak et al. [29] attempted to use two methods for detecting SFDIA. The first of the two models they proposed utilizes the multivariate Gaussian semi-supervised learning algorithm and the second model utilizes a distributed support vector machine (SVM) based algorithm, which requires no supervised learning. Both models use principal component analysis to reduce the dimension of measurements. They use a non-linear classifier with a Gaussian kernel to define the attacked and the safe modes' boundary. In their simulation to generate data using MATPOWER, they collect active power measurements from each transmission line. However, the dataset they use is too small, namely 1000. They report that the semi-supervised model can attain up to

82% and the SVM-based method can attain up to about 78% of F1-score. This work is a preliminary investigation of applicable of machine learning techniques for detecting SFDI attacks on SE in transmission systems, and they demonstrate that this approach has promise. They simulated a 118-bus system.

Foroutan and Salmasi [32] proposed a four-phase classification: 1) they reduced the dimension of the data using PCA, 2) used a positively labeled set to build mixed Gaussian model, 3) used a mixture dataset to choose a proper classification threshold and finally 4) used an unlabeled dataset for evaluation. They used an IEEE 118-bus system simulated using MATPOWER to validate their semi-supervised learning solution by comparing its performance with those of SVM and MLP. They reported having F1-score of 95.65% for their MGD-based method.

Ozay et al. [88] used semi-supervised SVM (S3VM), variation of supervised SVM, to detect SFDIA. S3VM is based on the assumption that the input data in the same cluster have the same labels and the difference in number of input data in sub-clusters is not large. However, since SFDIA is sparse, yielding an imbalanced dataset, S3VM does not work very well when sparsity is high. The authors conducted experiments on simulated IEEE 9- 57- and 118-bus systems.

Chakhchoukh et al. [19] proposed a detection method using a newly developed machine learning technique known as the density ratio estimation (DRE). The DRE [101] is an effective countermeasure against cyber-attacks, which does not require supervision or an attack model. Given two sets of samples from two different distributions, DRE learns the ratio between the two probability density functions. Here the authors use data from “normal” operation of the power system as one set of samples and the data that might be contaminated with noise, bad-data or false data injection attacks as another. They demonstrated the method using simulation of an IEEE 118-bus system.

They reported that the DRE method can detect 100% of the attacks. However, they did not report the corresponding FAR.

2.5 DETECTION USING REINFORCEMENT LEARNING

In reinforcement learning, the machine learning model uses an agent that facilitates learning in an interactive environment by trial and error using feedback from its own actions and outcomes. In reinforcement learning the goal is to find a suitable action model that would maximize the total cumulative rewards and punishments of the agent.

Kurt et al. [57] proposed a detection mechanism using state–action–reward–state–action (SARSA), a reinforcement learning algorithm. They formulated the problem of SFDI attack detection as a partially observable Markov decision process (POMDP). They tested their proposed solution using MATPOWER-generated data for an IEEE 14-bus system.

2.6 SUMMARY

In this chapter, a review of the works that use machine learning to detect SFDIA is presented. A few observations from this literature review include:

1. None of the works combined both supervised and unsupervised models together in one solution.

2. A few works mentioned here used multiple classifier models as individual models, but none used those classifiers together as ensemble models.

3. Different learning classes had been attempted by the researchers. However, no attempts to use adversarial learning or explanation-based learning is undertaken so far.

4. Almost all the works used simulated datasets for testing their models. Three works used power flow data from power grid, but they added synthetic attacks to the dataset later making it simulated data.

5. Most of the attack data are generated randomly. However, in real-life an adversary would craft the SFDIA intelligently considering the system dynamics. Unless tested with real attack data, the performance of the methods against such sophisticated SFDIA will remain unknown.

CHAPTER 3

State Estimation and Stealthy False Data Injection Attacks

3.1 INTRODUCTION

This chapter briefly describes state estimation in power transmission system and the mathematical formulation for stealthy false data injection attacks on static state estimation in power systems. The chapter also describes the SFDI attack process.

3.2 STATE ESTIMATION IN POWER TRANSMISSION SYSTEMS

State estimation (SE) at the transmission system in electric power grids is a key function in supervisory control and planning of the system. It is used to provide the best estimate of the values of the system's unknown state variables, i.e., voltage magnitudes and phase angles of the system buses, from the measurements available from the network model and sent by the SCADA system to the control center. State estimation is run in every few seconds to a few minutes. The functions of the state estimator include identifying and correcting anomalies in the data, suppressing any bad data, and refining the measurements. Finally it gives a set of state variables that is acceptable to the operator and as inputs to other computational programs of the energy management system (EMS). Figure 3.1 gives the data flow in a typical state estimator.

The SE process generates a "residual vector" which is analyzed to detect possible abnormal measurements by checking for residuals that do not obey the Gaussian assumption. While the standard residual analysis tests can identify the presence of errors, it may not detect "stealthy" FDIA because an attacker familiar with the power transmission system topology information can carefully craft the data amounts to be injected in a way that the residual of the original measurement vector remains the same as the residual of the measurement vector with the injected data.

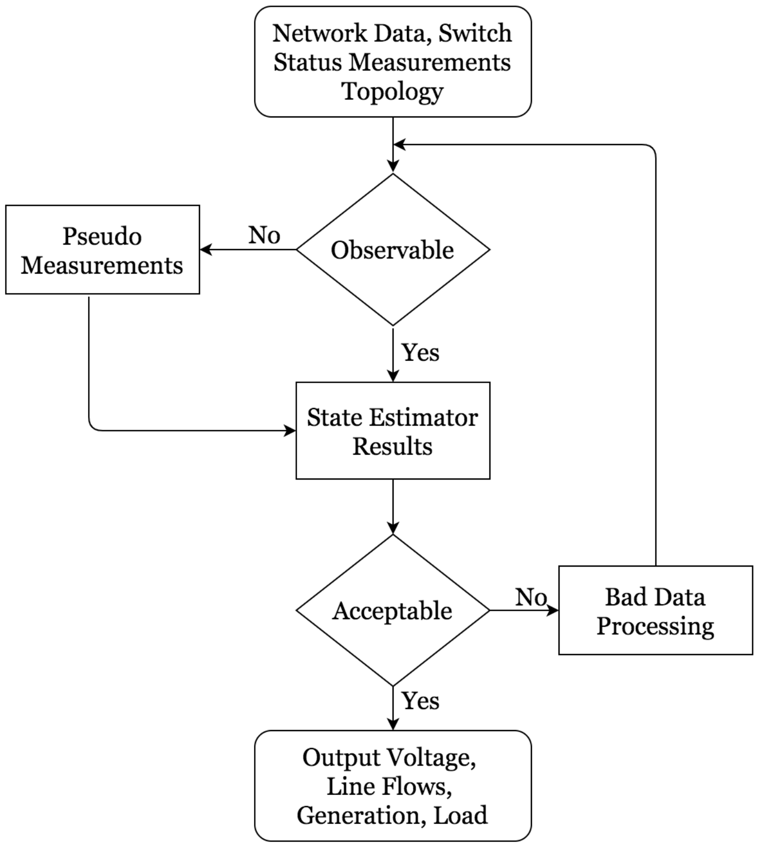

Network Data, Switch Status Measurements Topology
Observable
No
Pseudo Measurements
Yes
State Estimator Results
Acceptable
No
Bad Data Processing
Yes
Output Voltage, Line Flows, Generation, Load

FIGURE 3.1: A flow chart of the state estimation process (from [105]).

The following EMS tools are dependent on the state variable values estimated by the SE.

Contingency analysis involves performing efficient calculations of system performance from a set of simplified system conditions. Contingency analysis is one of the most important tasks of the Energy Management Systems (EMS). Used for the purpose of fast estimation of system stability right after outages, by bulk power system operators.

Unit commitment is an operational planning method used to determine a schedule called Unit Commitment Schedule which tells us beforehand when and which units to start and shut down during the operation over a pre-specified time, such that the total operating cost for that period becomes minimum.

Optimal Power Flow is a technique used to simulate load flow through an AC power system and find the combination of the flows that is operationally and economically optimal.

Locational marginal pricing tools are used to price out the cost of electricity for the local distributors or consumers.

3.3 FORMULATION OF STATE ESTIMATION AND FALSE DATA INJECTION ATTACKS

This section introduces the mathematical formulation for stealthy false data injection attacks on static state estimation in power transmission systems [2, 74].

In a power transmission system, the static SE is run after collecting measurements from the SCADA units at chosen time snapshots and communicating those to the control center every few seconds to a few minutes. These measurements are power flows and injections, as well as voltage magnitudes. The AC static SE estimates the (state) vector $\boldsymbol{x} \in \mathbb{R}^n$ that contains phase angles and voltage magnitude at the different buses, where $n = 2k - 1$ and k is the number of buses in the system. The slack bus phase angle is assumed to be the reference and is fixed to 0. The state vector obeys the following nonlinear equation:

$$\boldsymbol{z} = \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{e} \tag{3.1}$$

The nonlinear vector function $\boldsymbol{h}(\cdot)$ is computed from the grid topology and the parameters for transmission lines, transformers and other devices. The error vector $\boldsymbol{e} \in \mathbb{R}^m$ is assumed Gaussian with a covariance matrix R, where m is the number of measurements. The vector of measurements $\boldsymbol{z} \in \mathbb{R}^m$ contains communicated readings from SCADA units. The AC SE is executed using an iterative algorithm based on the weighted least squares (WLS) [2] to compute and estimate the vector $\boldsymbol{x}$, i.e.,

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_{k-1} + H_k^{\sharp} (\boldsymbol{z}_k - \boldsymbol{h}(\boldsymbol{x}_{k-1})) \tag{3.2}$$

where $H_k^{\sharp} = (H_k^{\top} R^{-1} H_k)^{-1} H_k^{\top} R^{-1}$, the matrix H_k is the Jacobian of $\boldsymbol{h}$ with respect to $\boldsymbol{x}$ at step k. The WLS algorithm is optimal under Gaussian noise.

Let $\Delta \boldsymbol{z}_k = \boldsymbol{z}_k - \boldsymbol{h}(\boldsymbol{x}_{k-1})$ be the k^{th} residual vector. After the convergence of the algorithm, i.e., once $\|\hat{\boldsymbol{x}}_k - \hat{\boldsymbol{x}}_{k-1}\| < \delta$ for some chosen threshold $\delta > 0$, the obtained residuals are analyzed by practitioners to detect possible abnormal measurements by checking for residuals that do not obey the Gaussian assumption. These abnormal or bad data could be due to natural failures such as sensor or communication misbehavior, or to FDI attacks. The most practical bad data detection rules are known as the chi-square test (χ_2) and the "3σ" rejection rule [2]. The iterative algorithm is equivalent to an estimation that is run iteratively after linearizing the regression in each step. The AC SE problem can also be reformulated as:

$$\boldsymbol{z} = H\boldsymbol{x} + \boldsymbol{e} \tag{3.3}$$

If contamination occurs due to an FDI attack then the measurement vector $\boldsymbol{z}$ received at the control center is replaced by $\boldsymbol{z}_a$ with $\boldsymbol{z}_a = \boldsymbol{z} + H\boldsymbol{c}$. The obtained new state is biased by the contamination vector $\boldsymbol{c}$, i.e., $\hat{\boldsymbol{x}}_a = \hat{\boldsymbol{x}} + \boldsymbol{c}$. The conventional methods detect such contamination by analyzing the residual (i.e., the difference between the measurement vector $\boldsymbol{z}$ and the calculated value from the state estimation, i.e., $\boldsymbol{z} - H\hat{\boldsymbol{x}}$). In the largest normalized residual test, if the largest absolute value of the elements in normalized residual is greater than a pre-defined threshold $\alpha > 0$, (α is generally chosen to be 3) the corresponding measurement is identified as bad data and reported to system operators. The measurement is removed and the estimation is re-executed.

3.4 STEALTHY FALSE DATA INJECTION ATTACKS

In the case of FDI attacks, if the injected data are large enough the conventional residual tests can detect them and these are called *non-stealthy FDI attacks*. In the non-stealthy case, the measurement matrix H is not known to the attackers and they simply generates random attack vectors and manipulate the meter readings.

On the other hand, if the attackers are familiar with the power system topology information or know the measurement matrix H, they can carefully craft the data amounts to be injected in a way that the residual $\boldsymbol{r}$ of the original measurement vector $\boldsymbol{z}$ remains the same as the residual $\boldsymbol{r}_a$ of the measurement vector $\boldsymbol{z}$ with the injected data $\boldsymbol{z}$.

$$\boldsymbol{r}_a = \boldsymbol{z}_a - H\hat{\boldsymbol{x}}_a = \boldsymbol{z} - H\hat{\boldsymbol{x}} = \boldsymbol{r} \tag{3.4}$$

These are called *stealthy FDI attacks* as they cannot be detected using the conventional methods based on residual analysis [74].

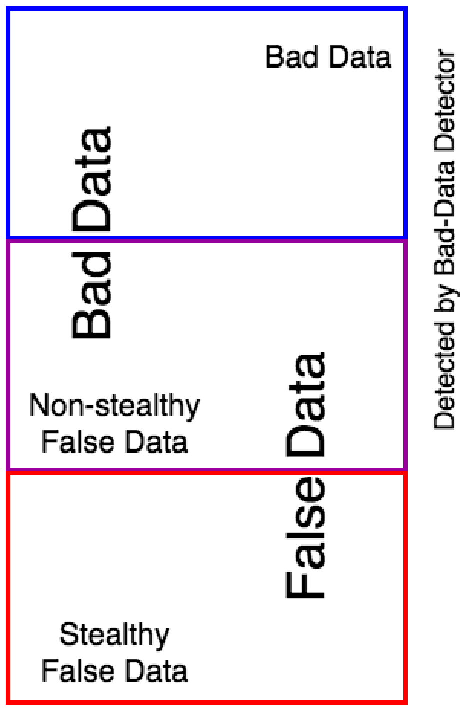


FIGURE 3.2: Diagram showing relationship between bad data, false data and stealthy false data.

Figure 3.2 shows the rough relationship between bad data and false data. The middle band can have both bad data and false data. The false data that fall into the middle band are detected by the state estimation's bad-data detector. The false data that fall in the bottom band, i.e., the stealthy false data, cannot be detected by the bad-data detector.

3.5 STEALTHY FALSE DATA INJECTION ATTACK PROCESS

False data injection attacks can be carried on different parts of the power grid, e.g., transmission systems, distribution systems, advanced metering infrastructure, etc. [72]. However in this research proposal, SFDI attacks only on the state estimation in the AC power transmission system are considered. Figure 3.3 shows a simplified diagram of

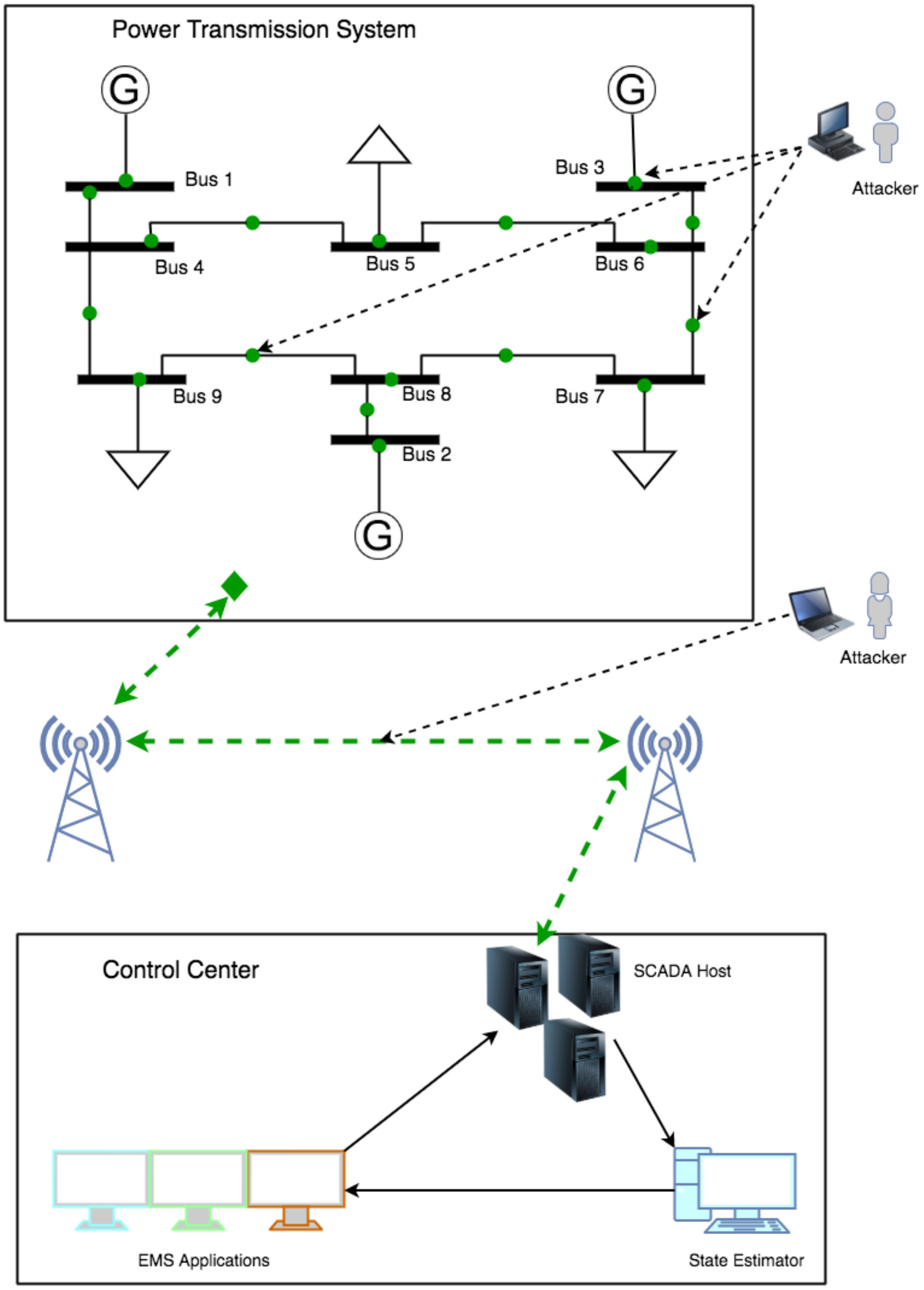


FIGURE 3.3: A simplified diagram of a power grid with its SCADA and communication systems showing the FDI attack vectors.

the power grid with the transmission system, the SCADA and the wireless communication links. It also indicates the possible attack vectors. An attacker can break into the remote sensors associated with the buses and modify the measurement data or/and compromise the communication channels, possibly through man-in-the-middle attacks, to intercept and modify the network packets.

The following are typical steps an adversary may follow for an SFDI attack:

1. Intrusion into the System (the stepping-stone):

   (a) If the adversary is an outsider, they will hack into the system using one or more of the usual cyber-attacks, e.g., spear-phishing, password-cracking, cracking the cryptographic protection, etc. or using a man-in-the-middle attack by compromising any wired or wireless communication channel.

   (b) An outside adversary can also be successful in installing a malware in the system either using the means above or using social engineering ploys. This malware may have the ability to steal system information, particularly the system topology.

   (c) If the adversary is a trusted insider, then s/he may already have the access and authority to get the system information.

2. Carry out SFDI attacks:

   (a) After the adversary has gained access into the system and obtained necessary system information, they can now surreptitiously change the measurement data and hence launch a stealthy FDI attack.

   (b) The operator and the state estimator assume that the data is correct and estimate the state variable values based on this false assumption.

(c) Since the state variable values do not represent the actual state of the system, calculation by any of the post-SE tools will be incorrect. This will cause adverse operation of the system and may result in malfunctions or major disruptions.

The goal of the attacker is to disrupt the operation of the transmission system leading to a failure in one or more component or bus, which may even trigger a cascade of failures, i.e., tripping of breakers because of power overload, and causing localized or wide-scale power failure.

3.6 SUMMARY

In this chapter, the static state estimation process for power transmission systems, the mathematics of bad-data in PTS, what constitutes SFDIA, and the SFDIA process are described. State estimation process calculates the values for bus system variables using the matrix of measurements from different RTUs. The bad-data detection tools can identify and rectify anomalous data and spoofed data. However, there is a class of spoofed data that cannot be detected by bad-data detectors. These stealthy false data injection attacks can be a second line of attack in a coordinated attack on the power grids.

CHAPTER 4 TAXONOMY OF FALSE DATA INJECTION ATTACKS

4.1 INTRODUCTION

Extensive research have been done since the identification of false data injection (FDI) attacks by Liu et al. [73]. Research on FDI attacks mainly focuses on three aspects: theoretical research, application research, and defensive research [62].

The theoretical research is concerned with developing the SFDI attack schemes, i.e., the construction of vectors to be injected into actual data matrices. These vectors, capable of evading detection by the control center, are developed for different components of the grid, situations, and constraints. In application research, the focus is on analyzing the impacts of SFDI attacks on power system operation, mainly on energy management systems (EMS) and market management systems (MMS), for instance, economic dispatch, congestion managements, etc. The defensive research is garnered towards proposing detection mechanisms and defense strategies from the viewpoint of the system operator.

In this chapter, a survey of the theoretical research and a taxonomy of SFDI attack models is presented.

4.2 RELATED WORKS

There are only a few works in the literature that attempted to classify false data injection attacks on smart grids. None of these works claimed to have developed a taxonomy of such attacks.

Guan et al. [37] in a limited survey of FDI attacks, described centralized and distributed attacks on the DC state estimation, a generic attack on AC state estimation, and a superficial discussion of FDI attacks on the control system. They did not cover many other FDI attacks and they limited their survey to only a few works.

Liang et al. [62] did a review of FDI attacks on the power grids. They summarized the different strategies of constructing valid FDI attacks under various constraints. They described FDI attacks when 1) attacker has full knowledge of the system, 2) attacker has incomplete knowledge of the system, 3) attacks are on the system topology, and 4) attacks are on the AC power flow. They did not cover attacks on other parts of the transmission system and on other parts of the power grid. Their work is a review of FDIA, rather a taxonomy.

Liu and Li [72] did a broader survey of FDI attacks covering attacks on the transmission system, distribution system and microgrid. However, the paper is more of a survey of FDIA models, impact analyses and defense strategies, than a taxonomy of FDIA.

In comparison to the previous works, this is an effort to present a taxonomy of attacks classified based on the power grid components targeted and based on the kind of impacts the attacks cause.

4.3 COMPONENT-BASED ATTACK CLASSIFICATION

In the following few sections, a taxonomy of FDI attacks based on the targeted components of the power grid is presented. Section 4.4 presents FDIAs on transmission systems, Section 4.5 on distribution systems, Section 4.6 on microgrids and Section 4.7

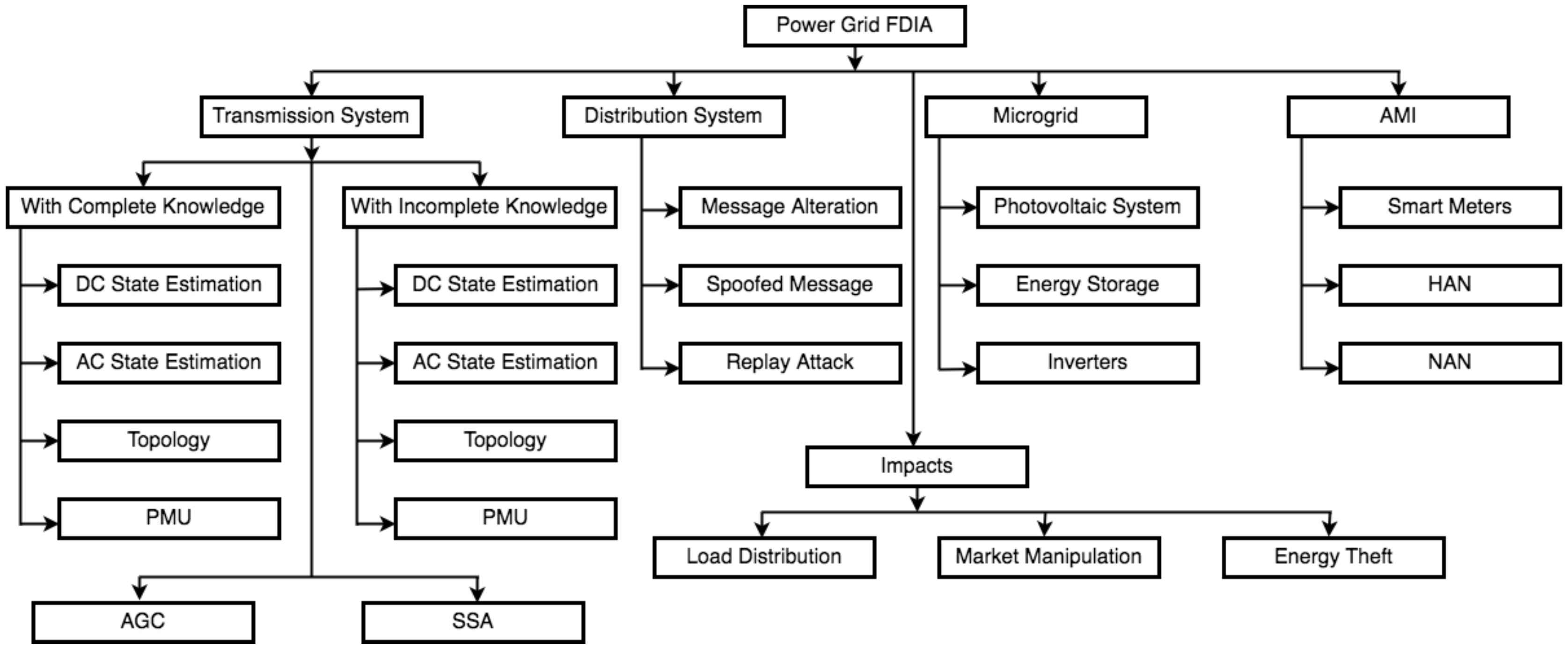


FIGURE 4.1: Diagram showing taxonomy of stealthy false data injection attacks on the power grid.

on advanced metering infrastructure (AMI). Figure 4.1 shows a diagram of a taxonomy of FDI attacks on the power grid.

4.4 ATTACKS ON TRANSMISSION SYSTEM

False-data injections in power grid was identified as cyber-attacks by Liu et al. [73] for the direct current (DC) power flow model in the transmission system. They argued that if an attacker can construct a false data injection attack such that the overall residue of the system will not increase, the attack on measurements can bypass the residual test for bad-data detection and constitute a successful attack. Rahman and Mohsenian-Rad [81] formulated the false-data injection attacks for the alternating current (AC) case which is more complicated than the DC case.

The FDI attacks on the transmission system can be broadly categorized in two groups depending on whether the attacker has full knowledge of the system topology and other system values or not.

4.4.1 *With Complete System Knowledge*

If an adversary has complete knowledge of the system configuration information, including system parameters, power grid topology, state estimation algorithm, bad-data detection mechanism, and transmission-line admittance values, etc., and has the ability to modify data at all the meters, they can adjust the false data injection attack vector in a way that the attack remains undetected and successfully passes the residue-based bad-data detection tests.

4.4.1.1 *Attacks on the DC State Estimation*

After the initial identification by Liu et al. [73] for the DC model, a number of works followed. Kosut et al. [56] proposed a graph-theory based algorithm to construct false-data injection attacks. Kim et al. [55] proposed a data-framing FDI attack where it was shown that an attacker can inject data in such a way that the energy management system (EMS) identifies correctly functioning measurement devices as a false data injection source. Yang et al. [123] proposed a collective sparse attack strategy where state variables in the same cluster are attacked by the same attack vector. Wang and Ren [111] studied the situation when FDIA is on transmission system and the topology is not fixed. Assuming that only one line can be cut off per time interval, they developed necessary and sufficient condition under which successful stealthy FDI attack can be generated.

4.4.1.2 *Attacks on the AC State Estimation*

Hug and Giampapa [48] analyzed stealthy false data attacks against the AC state estimation and showed that an attacker can launch a successful stealth attack if they have the full network information of a power grid. They developed a graph-theoretic approach to minimize the number of measurements to be modified for attacking a load measurement at a bus. Since the injected power at a zero-injection bus must be zero to ensure power flow feasibility, any malicious modification of the readings of these measurements will be detected with a relative high probability. Thus they concluded that the zero-injection buses can improve the resilience of a power system to false data injection attacks.

Chakhchoukh and Ishii [18] identified two attack scenarios on both AC and DC state estimations. In one scenario, called *masked attacks*, attacks are possibly unidentified with a convergence to an arbitrary state unknown to the attacker. In the second scenario, called *stealthy attacks*, convergence to a known state is targeted by the attacker.

4.4.1.3 *Attacks on the Topology*

In topology-based attacks, adversaries aim at creating errors in the topology information by virtue of knowledge of measurement configuration. In power transmission systems, the status of each transmission line, whether the line is in service or not, is sent to the control center in real time. Under normal operations, the topology of a transmission system changes due to faults or forced outages of transmission lines. However, during transmission of this status data, an attacker can intercept and modify the status of a line sent to the control center exploiting the vulnerability of communication networks. This alteration in the status data can lead the control center to erroneously believe the system is operating under a topology different from that in reality. This can lead to serious implications: a grid that is under stress may appear to be normal to the operator thereby delaying the deployment of necessary measures to ensure stability. On the other hand, an operator may deem a normally-operating grid to be under stress and can take costly remedial actions including load shedding. To construct stealthy attacks, the attacker modifies the meter data for transmission systems and the network topology data simultaneously in such a way that the estimated topology is consistent with the modified data.

Kim and Tong [54] derived a necessary and sufficient condition for the existence of an undetectable attack for strong adversaries who has knowledge of the complete system, in other words can observe all meter and network data. They presented a simple undetectable attack, called *state-preserving attack*, where the attack intentionally preserves the state in order to have a sparse attack vector. Under this attack an adversary can simulate the physical outage of a transmission line without actually severing the line where a pair of additional power increments is injected into the power measurements (and not the phase angles) at the terminal buses of the targeted line.

The topology attack model in Kim and Tong [54] assumes that the injected false power at a bus is infinite which is quite impractical since a control center operator usually has some knowledge about the load distribution of a power grid and can predict future loads using load forecasting. To overcome this drawback, Liu and Li [71] proposed a heuristic algorithm to reduce the required network information for determining a feasible attack region, thereby a local topology attack model that limits the injected power at a bus within a certain range.

Liu and Li [71] and Li et al. [60] proposed a topology attack model to mask the physical outage of a transmission line injecting false data into some measurements such that the new power flow is consistent with the case in which the line is on outage. This will cause the control center to believe that this line is still in service although it is physically on outage.

4.4.1.4 *PMU Attacks*

Phasor measurement units (PMU) are used to estimate real-time phasor and sequence measurements from geographically dispersed nodes in the power transmission grid with high-sampling rate. These measurements, known as synchrophasors, including magnitude and phase angle of an electrical phasor quantity, such as voltage or current, are time-synchronized to an absolute time reference provided by the global positioning system (GPS). These measurements are transmitted to the control centre and are used to reliably evaluate, monitor, control, and protect the grid.

Use of GPS allows PMUs to synchronize real-time measurements of multiple remote points on the grid with high accuracy. However, dependence of synchronized measurements on GPS signals has been identified as an attack surface. Two kinds of attacks have been identified. Zhang et al. [131] and Jiang et al. [50] demonstrated that PMUs are vulnerable to *spoofing attacks* that provide false time stamps on the PMU measurements. The spoofed time stamps cause two types of errors: the phase angle error and the time-of-arrival (TOA) error. The phase angle error will render the operator unable to or to wrongly detect the outages of transmission lines; the TOA error will result in miscalculation of the locations of disturbance events. The spoofing can also attack the clock offset of a PMU to disrupt voltage stability monitoring.

Liu and Li [68] showed that the line outage detection using the PMU data can be significantly disrupted by false data injection attacks. The masking scheme is to maximize the residual of the outage line in the detection algorithm. When a line outage occurs, the operator calculates the expected phase angle changes due to this outage based on received real-time measurements and compares the values to these real phase angle changes measured by PMUs to locate the fault line. With the PMU measurement data changed, the operator is forced to take incorrect remedial actions causing disruptions.

4.4.2 *With Incomplete Network Information*

So far the review dealt with FDI attacks that are based on the assumption that the attacker knows the complete configuration information of the power grid and power network. In practice, it is difficult for an adversary to know the complete network information and therefore it is not possible to launch a successful FDI attack on power grids. However, research has established that it is still possible for an adversary to launch a successful FDI attack even if they have incomplete information about the grid. In this section, a review of research that developed FDI attack models with incomplete system information will be presented. The focus of this line of research is to ascertain enough system information from grid operation to launch a successful FDI attack.

4.4.2.1 *Attacks on DC State Estimation*

Esmalifalak et al. [30] observed that topology information is embedded into the correlations among power flow measurements when the system parameters (e.g., active or passive loads) vary within a small dynamic range, and that the topology information can be extracted from this. In practice, power system topology remains the same unless there is a reconfiguration of the system. Hence, due to the slow dynamic nature for a short period of time, the equivalent knowledge of the topology can be revealed using the correlations among power flow measurements. They proposed an independent component analysis (ICA) based algorithm to obtain topology information from power flow measurements.

Rahman and Mohsenian-Rad [93] formulated FDI attacks with incomplete information from both the attacker's and system operator's viewpoints and introduced a novel vulnerability measure that can compare and rank different power grid topologies against such attacks. Rahman and Mohsenian-Rad [93] and also Giani et al. [34] developed an approach to construct FDI attacks by splitting the grid into several subnetworks, and increasing or decreasing phase angles by the same amount incrementally in each subnetwork. Since the bus angles in the subnetworks change by the same degrees, the power flows in the area covered by the subnetworks will not change. They termed this as “local attack” where the modified phase angles constitute the attacks.

Kekatos et al. [52] showed a way to identify grid topology using electricity prices. They proposed an iterative alternating direction method of multipliers (ADMM) based algorithm to estimate the grid Laplacian matrix from the locational market prices (LMPs) using a regularized maximum likelihood estimator, since under the DC model, LMPs correspond to the Lagrange multipliers of network-constrained economic dispatch problem. The authors solved the problem using an ADMM-based algorithm and demonstrated that the estimated Laplacian matrix is close to the real topology. Moya et al. [84] introduced the concept of correlation index for wide-area measurement attacks at sub-transmission level inspired by forensic analysis. The correlation index was developed by studying the pattern of measurement attacks.

Yang et al. [123] developed three attack models, namely *least-effort attack model*, *optimal attack model* and *optimal attack model for large grid*. First an adversary can launch the least-effort attack in order to find the sparsest attack vector so that the attack can be conducted by compromising the minimal necessary number of sensors. After gaining knowledge of some state variables, the adversary can launch the linear transformation based optimal attack to launch a full-fledged FDI attack on the grid. The third model is based on a heuristic algorithm that can be used to launch a full-fledged FDI attack on a large grid network.

Yu and Chin [124] proposed a method to generate sparse false data injection attacks of the general kind on both the DC and AC cases by applying the principal component analysis (PCA) to the measurement data matrix into a new subspace, preserving the spatial characteristics as much as possible. PCA is a multivariate statistical technique which can transform the correlated observations into uncorrelated variables known as principal components. These orthogonal principal components are the linear combinations of the original observations. The authors construct FDI attacks based on the data of the new projected space. These attacks are generated without the knowledge of the measurement matrix and the assumption regarding the distribution of state variables. In order to generate the false data injection attack vector, they applied PCA on the vector of actual measurements, and multiplied it with the transpose of a vector with a non-zero vector obtaining the attack vector.

Anwar et al. [8, 9] extended the idea of Yu and Chin [124] to construct an attack from only the measurement signals even in the presence of gross, non-Gaussian, errors. They further developed a sparse optimization based blind (no system knowledge) stealthy attacks construction strategy that is stealthy and successful even in the presence of grossly corrupted measurements due to device malfunction and communication errors.

4.4.2.2 *Attacks on AC State Estimation*

AC power flow equations are highly nonlinear and it is much more difficult to formulate an attack for the AC model with incomplete network knowledge. Rahman and Mohsenian-Rad [94] proved that constructing FDI attacks on AC systems using DC models will be easily detected by the bad-data detector.

Liu and Li [70] extended the local attack model for DC systems in [67] to the AC case by replacing the estimated voltage magnitudes at the boundary buses in the attacking region with the corresponding voltage measurements and setting the power flows on the tie-lines calculated using the estimated voltage magnitudes and phase angles to be the corresponding line flow measurements on these tie-lines.

4.4.3 *Attacks on Topology*

Kim and Tong [54] proposed a heuristic method for constructing FDI attack on network topology for the case when the adversary has only limited knowledge of the system. They demonstrated that the physical outage of a line can be simulated by launching the *state preserving attack* (described in Section 4.4.1.3) without physically disconnecting this line. Kim and Tong [54] also demonstrate that a topology attack has significant impact on the locational marginal price (LMP).

4.4.4 *Attacks on Automatic Generation Control*

Tan et al. [102, 103] formulated stealthy false-data injection attacks on the sensor measurements for automatic generation control (AGC), a fundamental control system used in power grids to maintain the grid frequency at a nominal value. The authors show that, if an attacker can gain access to sensor data and a few system constants, they can learn the attack impact model and achieve the optimal attack. False data injection attacks on the sensor measurements for AGC can cause frequency excursion that triggers remedial actions, by control center operators, such as disconnecting loads or generators, that may lead to blackouts and potentially costly equipment damage.

4.4.5 *Attacks on Static Security Assessment*

Chen *et al.* [21] investigated the impact of FDI attacks on the static security assessment (SSA), a process that determines if the power system is in a secure or alert (insecure) state; the secure state implies the load is satisfied and no limit violation will occur under present operating conditions and in the presence of unforeseen contingencies (i.e., outages of one or several lines, transformers or generators). They showed that successful FDI attacks can manipulate the results of SSA. They identified two scenarios. In *fake secure signal attack*, the system operator is made to believe that the system is operating in a secure condition when actually it is not. In *fake insecure signal attack*, the operator is deceived to make corrective actions, such as generator rescheduling, load shedding, etc. when these actions were not warranted. f

4.5 ATTACKS ON DISTRIBUTION SYSTEMS

The existing research on FDI attacks against state estimation in transmission systems cannot be trivially extended to distribution systems. In order to launch an FDI attack on the distribution system, the attacker must know the estimated state of the system. This makes the FDI attacks difficult to be implemented in practice.

Lim et al. [63] was among the first to investigate false data injection attacks in the power distribution systems, even though they didn't call it FDI attacks. FDI attacks on the distribution systems are realized by tampering with the messages between the server and the feeder remote terminal units (FRTU). These messages contain voltage and current data and control commands. The authors categorized the attacks into three distinctive attacks. First, altering the contents of these messages and sending to the FTRUs. The altered messages can control automatic switches in the system maliciously, eventually causing power outages. Second, creating spoofed messages and injecting them in the communication channel, thereby delivering illegal commands to the FRTUs. Third, sending a previous message again, thus effectively delivering incorrect time-varying information that reflects system status and actions. This is a replay attack.

Guo et al. [38] showed that an attacker on the distribution system can attack the feeders or FRTUs, subsystems or the customers. Deng et al. [27] showed that an adversary can approximate the system state based on the power flow or injection measurements and construct attacks on the state estimation of the distribution system.

4.6 ATTACKS ON MICROGRIDS

Microgrids are local and decentralized low-voltage electric power grids that can disconnect from traditional power grid for autonomous and self-sufficient operations.

Lin et al. [64] investigated FDIA against the distributed energy routing process in microgrids. They considered several general attacks, in which the adversary may manipulate the quantity of energy supply, the quantity of energy response, and the link state of energy transmission. They quantitatively demonstrated that the false data injected by those attacks cause imbalanced demand and supply, increase the cost for energy distribution, and disrupt the energy distribution.

Zhang et al. [128] investigated the impacts of FDIA on the dynamic microgrid partition process. They identified three FDI attacks on microgrids: i) claim less energy than what can be provided, ii) claim more energy than what is required, and iii) both i) and ii) at the same time. They show that for all these situations injected false data can disrupt the dynamic microgrid partition process.

Hao et al. [40] observed aberrant operation of a microgrid when false data is injected into the voltage controller of the substation. Liu et al. [71] proposed an FDI attack model against control signals of solar photovoltaic (PV) and energy storage system (ESS) controls in microgrids, Li et al. [59] studied FDIA against the control signals of the inverters in microgrids, and Beg et al. [14] considered false data injection attacks in cyber-physical DC microgrids.

Zhang et al. [127] investigated impact of FDIAs on the economic vulnerabilities in microgrids by categorizing the attacks by their utilization levels and monitored the stability of microgrids under different conditions.

Liu et al. [66] considered FDI attacks on communication links in a microgrid and theoretically formulated the conditions when the data injection constitutes i) a false attack, and ii) a missed or undetected attack. They further defined two classes of undetectable attacks: i) zero trace undetectable, and ii) non-zero trace undetectable.

4.7 ATTACKS ON ADVANCED METERING INFRASTRUCTURE

Advanced metering infrastructures (AMI) facilitate bidirectional communication between smart meters located at the consumers and utility companies, allowing information about consumption, outages, and electricity rates to be shared reliably and efficiently and for efficient, accurate and advanced monitoring and control.

Grochocki et al. [36] investigated FDI attacks as part of larger scheme of cyber-attacks on smart meters, neighborhood area networks (NAN) and home area networks (HAN). They developed an elaborate attack tree after performing threat analysis. FDI attack is an important part of this attack tree.

Anwar et al. [10] described a relationship between the power system stability indices and the FDI attacks when the attacks are injected through smart meters. They identified the level of vulnerabilities of each smart meter in terms of different degrees of FDI attacks.

4.8 IMPACT-BASED ATTACK CLASSIFICATION

4.8.1 *Load Redistribution Attacks*

Yuan et al. [125] developed the concept of load redistribution (LR) attack that can disrupt the grid operation by attacking the security-constrained economic dispatch (SCED). The SCED minimizes total system operation cost (e.g., generation cost, load shedding cost, etc.) by re-dispatching the generation outputs. In order to achieve a successful LR attack, the attacker makes sure that the attack amount at a load measurement must be limited within a small range to reduce the situational awareness of the system. Under an LR attack, the falsified SCED solution leads the operator to redistribute the loads in a way that may drive the system to an uneconomic operating state. The damage of LR attacks to power system operations can manifest in an immediate or a delayed fashion. The authors classified the LR attacks into i) immediate attack and ii) delayed attack. Yuan et al. [126] further developed the concept of LR attack by developing models to quantify the damage of both immediate and delayed attacks.

Liu and Li [69] and Liu et al. [67] investigated local load redistribution attacks with incomplete information. Xiang et al. [115] demonstrated that the LR attacks have a non-negligible impact on the power system reliability. The authors modeled the FDI attacks using semi-Markov modeling for both non-encrypted and encrypted communication and quantified the influence of load redistribution (LR) attack on the long-term power supply reliability. Xiang et al. [116] showed that LR attacks can be part of a coordinated cyber-attacks on the power grid. Mohsenian-Rad and Leon-Garcia [81] showed that the direct load control command signals and indirect load control price signals can be attacked by launching load altering FDI attacks.

4.8.2 *Attacks on the Electricity Market*

FDI attacks can be constructed to manipulate the locational market pricing (LMP) by modifying the measurement matrix of the state estimation, because the real-time LMP is calculated using estimated congestion pattern obtained from the state estimation. Xie et al. [117, 118] were the first to study this kind of attacks on the electricity markets and formulated a convex optimization problem to construct an optimal attack vector to maximize profit in the real-time market. They also developed a heuristic-based method for the attacker to improve the attack efficiency.

Jia et al. [49] took into consideration the effects of both the RTU measurements and the system topology state on real-time LMP and proposed formulation to calculate the injected false-data vectors for causing congestion pattern that affect the LMP most. Choi and Xie [22] studied the impacts of topology errors on LMPs. Esmalifalak et al. [31] studied the effect of FDIA on power market using a game-theoretic approach. Rahman et al. [95] proposed a formal verification-based framework to analyze the impact of topology attacks on the integrity of optimal power flow (OPF) and, hence, electricity pricing since the LMP is a by-product of OPF.

4.8.3 *Energy Deceiving Attacks*

It was seen before that the smart meters are susceptible to FDI attacks and an adversary can cause a major disruption in the system by attacking a set of critical meters. McLaughlin et al. [78] demonstrated how energy theft can be achieved by altering readings of a set of smart meters.

While investigating FDI attacks against the distributed energy routing process in microgrids, Lin et al. [64] identified the energy deceiving attack. The authors showed how an attacker can manipulate the memory of smart meters in the microgrids and inject erroneous energy demand and supply messages to cause energy deceiving attacks.

4.9 SUMMARY

False data injection attacks or covert data integrity attacks can be targeted on different parts of the smart grid, and for different purposes. This chapter presented a comprehensive review and a taxonomy of different FDI attack models classified based on the components attacked and also the impacts intended.

CHAPTER 5

Security Analytics Framework

5.1 INTRODUCTION

In this chapter, the security analytics framework that is used to detect SFDI attacks on state estimation in power transmission systems is presented. Security analytics has been around for just about a decade, mostly to protect from threats in the cyber systems. In this dissertation research, security analytics is being used to detect cyber-attacks in a cyber-physical system.

5.2 SECURITY ANALYTICS

Security Analytics is a data-driven approach to cybersecurity focused on the analysis of data using machine learning to produce proactive security measures, including detection, prevention and defense mechanisms. For example, monitored network traffic could be used to identify indicators of compromise before an actual threat occurs. Security analytics often, but not always, employs big data analytics.

In this dissertation research, a security analytics framework is used for detecting SFDI attacks. The central part of the framework is to use machine learning models for attack detection where machine learning models are first developed by letting the models learn the data behavior, i.e., relations in different types of data instances, through the training process. Once the models exist, real-time data is classified as either legitimate or anomalous in a detection phase. A key disadvantage of using machine learning is that the training process typically requires significant time and computational resources. However, once the model is trained, subsequent analysis is generally efficient. In this framework, a number of supervised, unsupervised, and stacking ensemble models are used.

The components of the security analytics framework are:

1. Risk Analysis
2. Threat Modeling
3. Data Generation
4. Data Engineering
5. Feature Reduction
6. Machine Learning Model Selection
7. Deployment of Trained Models
8. Monitoring for Attacks

A process flow diagram of the scheme is given in Figure 5.1. The process flow starts with a brief risk analysis, followed by threat modeling where the attack models are identified. These attack models are used when data generation module creates the synthetic datasets. Data engineering normalizes the data with standard scaling and balances the datasets for the supervised models. The scaled and balanced datasets are used by the feature reduction module. All the machine learning methods separately use the feature-reduced datasets to train. The trained models are tested and evaluated using standard evaluation metrics. This model selection phase produces the best performing models which are to be deployed to detect SFDI attacks. The deployed models

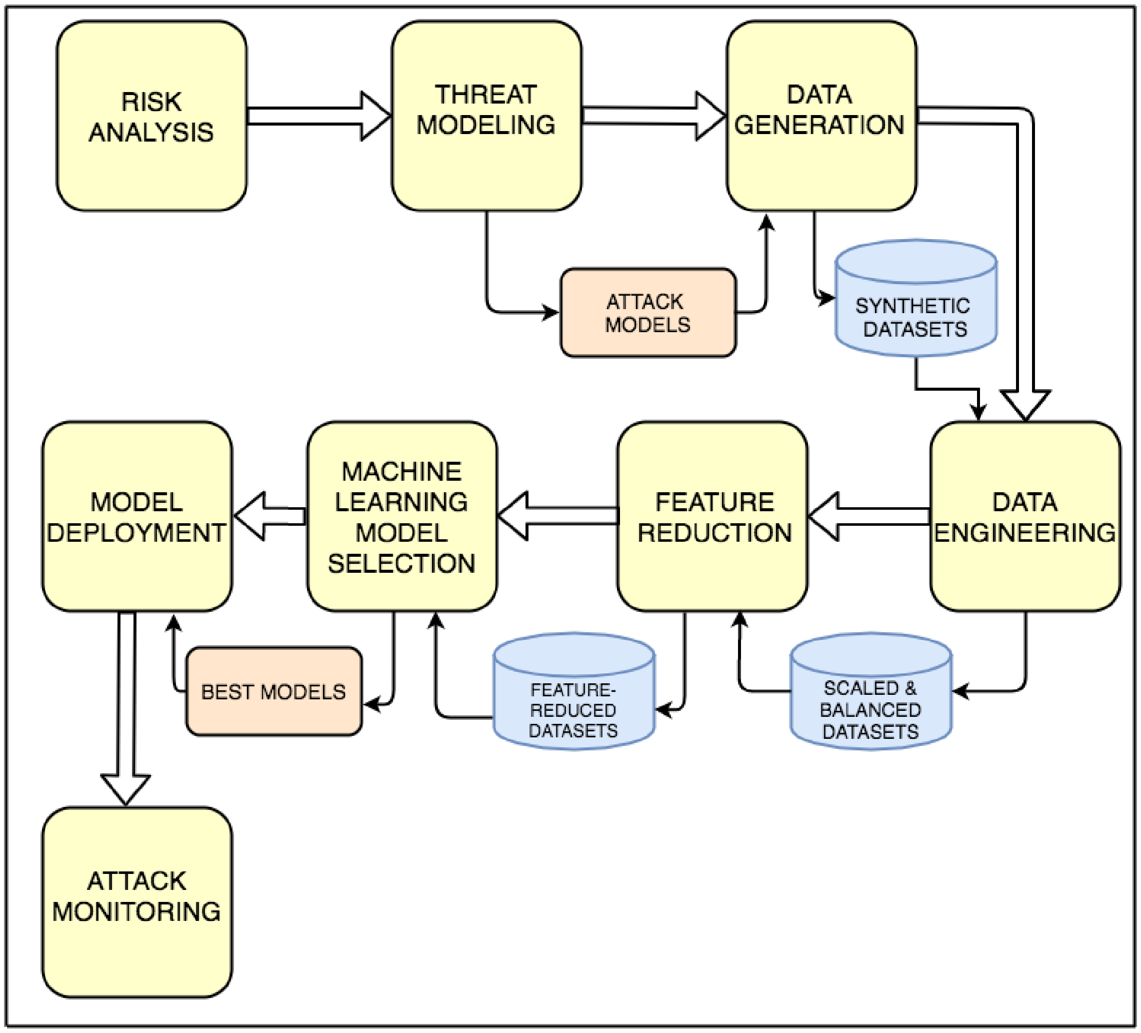


FIGURE 5.1: A process flow diagram of the security analytics framework.

monitor and analyze the oncoming SE measurement data vectors for the presence of any SFDI attack.

The following sections describe the components of the security analytics framework.

5.3 RISK ANALYSIS

Risk is the level of impact on the organization's operations, operational assets, and personnel given the potential impact of a threat and the likelihood of the threat occurring. In other words, risk is the possibility of damage or harm and the likelihood that damage and harm will be realized. Risk analysis is an effort to identify vulnerabilities and their related threats, assess the potential costs of exploitation, and determine appropriate and cost-effective security controls [113].

In this dissertation, it is considered that SFDI attacks have already been identified as threats to the power transmission systems, since it is assumed that the adversaries have already intruded into the system by way of compromising sensors or communication channels and have adequate knowledge of the system topology to craft effective SFDI attacks. As shown in Section 4.8, the impacts of the SFDI attacks on the system depend on the particular kind of SFDI attack being constructed. A risk analysis of SFDI attacks will have to consider the type, scope, and intensity of the attacks, the assets and services affected, the duration of the affected services, severity of physical, economic, and intangible damages, etc. Risk analysis of SFDI attacks on state estimation in power transmission system is out of scope for this dissertation research.

5.4 THREAT MODELING

A threat is an agent that can potentially compromise the normal operation of a system. The threat agent can be a human actor, or an event or circumstance. The agent can be a malicious actor or a benign but naive actor accidentally causing a security incident. Threats are identified, categorized, and prioritized using the threat modeling process.

For the case of SFDI attacks, the threat agents are malicious, purposeful, and highly resourceful. In this limited threat modeling, the different ways adversaries can constitute attacks on the measure data are identified. These attack models are described below.

The threat model is built on the following assumptions:

1. The attacks are only on the static state estimation measurement data in AC power transmission systems.
2. The attacks are targeted attacks and not random attacks. That means the same set of compromised RTUs are attacked for the entire duration of the attack.
3. The attacker has partial knowledge of the system topology. They know the regions in vicinity of the compromised RTUs.
4. The attacks are general attacks without intending any particular impact.

5.4.1 *Attack Models*

As described in Chapter 4, the SFDI attacks can occur at different points of the transmission system enabling the attackers according to the access point. An attacker can break into one of the remote sensors associated with the buses and modify the measurement data or compromise the communication channels at the control center or the Enterprise SCADA center, possibly through man-in-the-middle attacks, to intercept and modify the network packets. In order to cover this broad scope of SFDI attacks, three attack models for the investigation in the proposed research are considered.

These attack models are described with the help of Figure 5.2. It shows a simplified diagram of a 14-bus transmission system, along with the cyber network that consists of the SCADA and the wireless communication links.

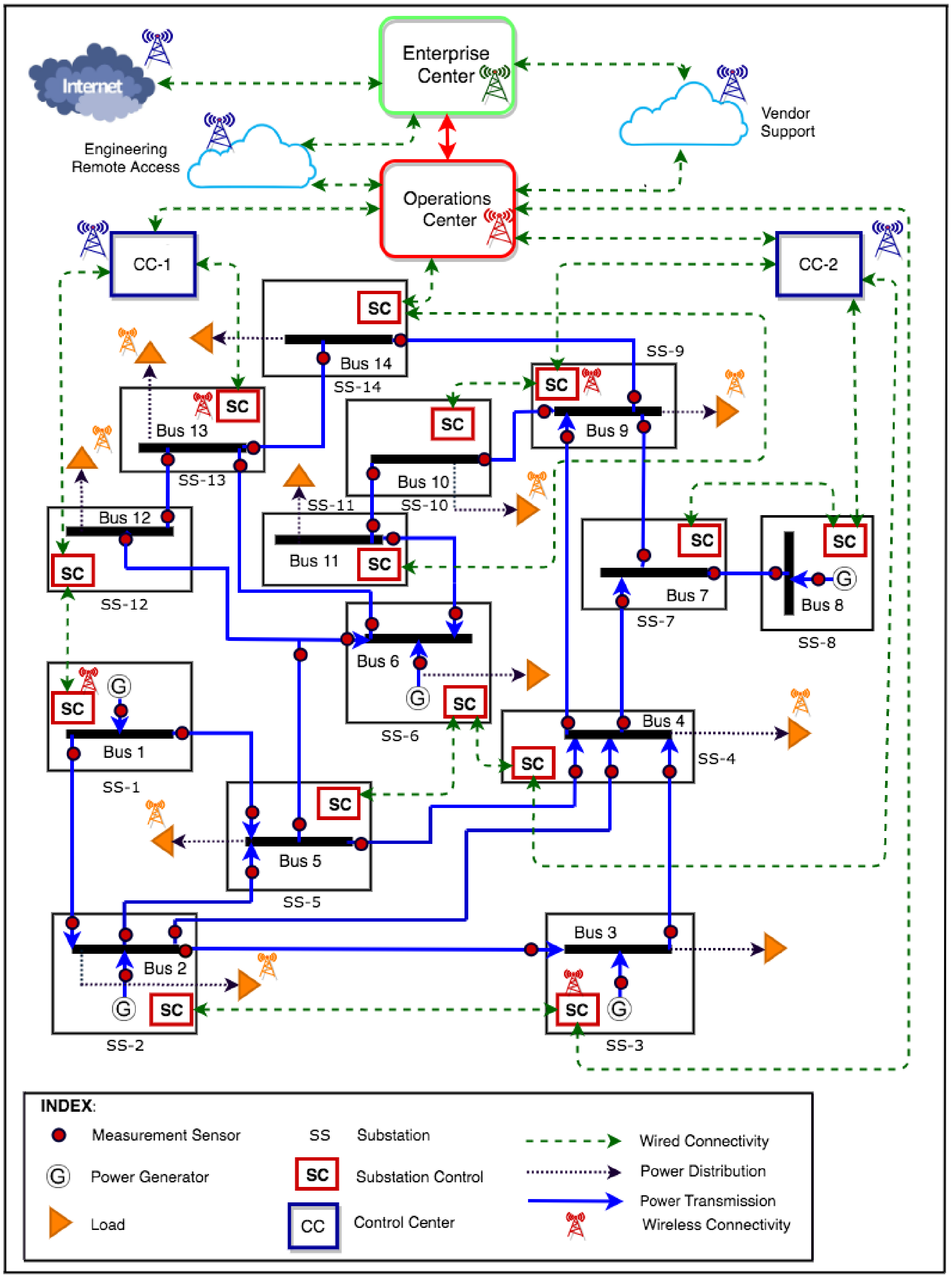


FIGURE 5.2: Representation of a 14-bus based power transmission system showing physical and cyber networks (adapted from [51]).

5.4.1.1 *Attack Model 1: Compromise One Substation*

In this scenario, it is assumed that sensors in one of the substations in the system are compromised by an attacker and hence the attacker can change the measurement values corresponding to the sensors in that Substation only. For example, if Substation SS-1 in Figure 3.1 is hacked then values from all the sensors in that substation can be modified.

5.4.1.2 *Attack Model 2: Compromise One Control Center*

A control center in the system collects the measurement values from the sensors in a number of buses, collates those and relays to the central SCADA center. If an attacker can intrude into one of the control centers, they can modify the measurements values that go through this control center. Referring to Figure 5.2, if the attacker gets access into control center CC-1, they can modify the measurement values from sensors associated with the buses in Substations SS-9, SS-6 and SS-5.

5.4.1.3 *Attack Model 3: Compromise Enterprise Center*

This scenario covers the cases when an outside adversary gets access into the enterprise SCADA center or when the attacker is a trusted insider hereby having full access to the matrix of measurement values. This later scenario can also be a malware that has gone into the SCADA center and has the ability to change the measurement values. In this model, since the attack happens at the central SCADA center, the adversary has the capability to change any or all the measurement values from all the sensors.

5.5 DATA GENERATION

For this research, instead of collecting, data was generated because of lack of availability of real data. Real SFDIA data, even if they exist, will be very few, because SFDI attacks are currently limited to the academic research labs.

The generation of data for this research is a two-step process: 1) State Estimation (SE) measurement data representing normal operation of the power system and 2) SFDI attack generation and injection into the measurement data.

5.5.1 *State Estimation Measurement Data Generation*

For generating power flow SE measurement data, standard IEEE 14-bus and 57-bus systems were simulated using MATPOWER [132]. The measurements are obtained from solving power flows using the MATPOWER and adding Gaussian measurement noise. The loads are considered to vary randomly around their average values.

The IEEE 14-bus system has 5 generators and 11 loads [106], as shown in Figure 5.3. The measurements are: 40 active power-flows, 14 active power-injections, 40 reactive power flows, 14 reactive power-injections, and 14 voltage magnitudes giving a total of 122 measurement features. The IEEE 57-bus system has 7 generators and 41 loads [106], as shown in Figure 5.4. The measurements for the 57-bus system are: 188 active power-flows, 57 active power-injections, 188 reactive power flows, 57 reactive power-injections, and 57 voltage magnitudes giving a total of 546 measurement features.

5.5.2 *Attack Generation*

The attack generation program is written using MATLAB. The buses considered compromised are selected randomly following the three attack models described earlier. Then the measurement values corresponding to those buses are replaced by attack data generated by this program. 10% of 'normal' data were randomly chosen to be 'attack' data.

The generated measurement vectors with attacks, i.e., modifications in the measurements, were used for state estimation using state estimation simulation in MATPOWER. As the state estimation successfully processed the data vectors, it demonstrated that bad-data detector could not detect the attacked data vectors. This shows that the generated attack data was stealthy.

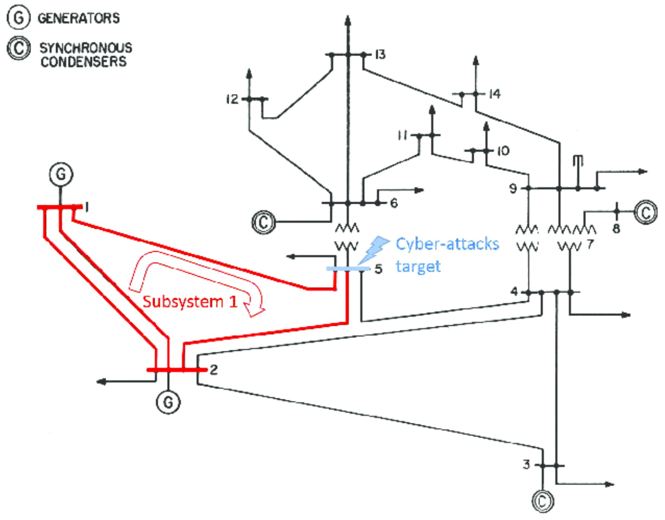


FIGURE 5.3: Diagram of standard IEEE 14-bus system (adapted from [106]).

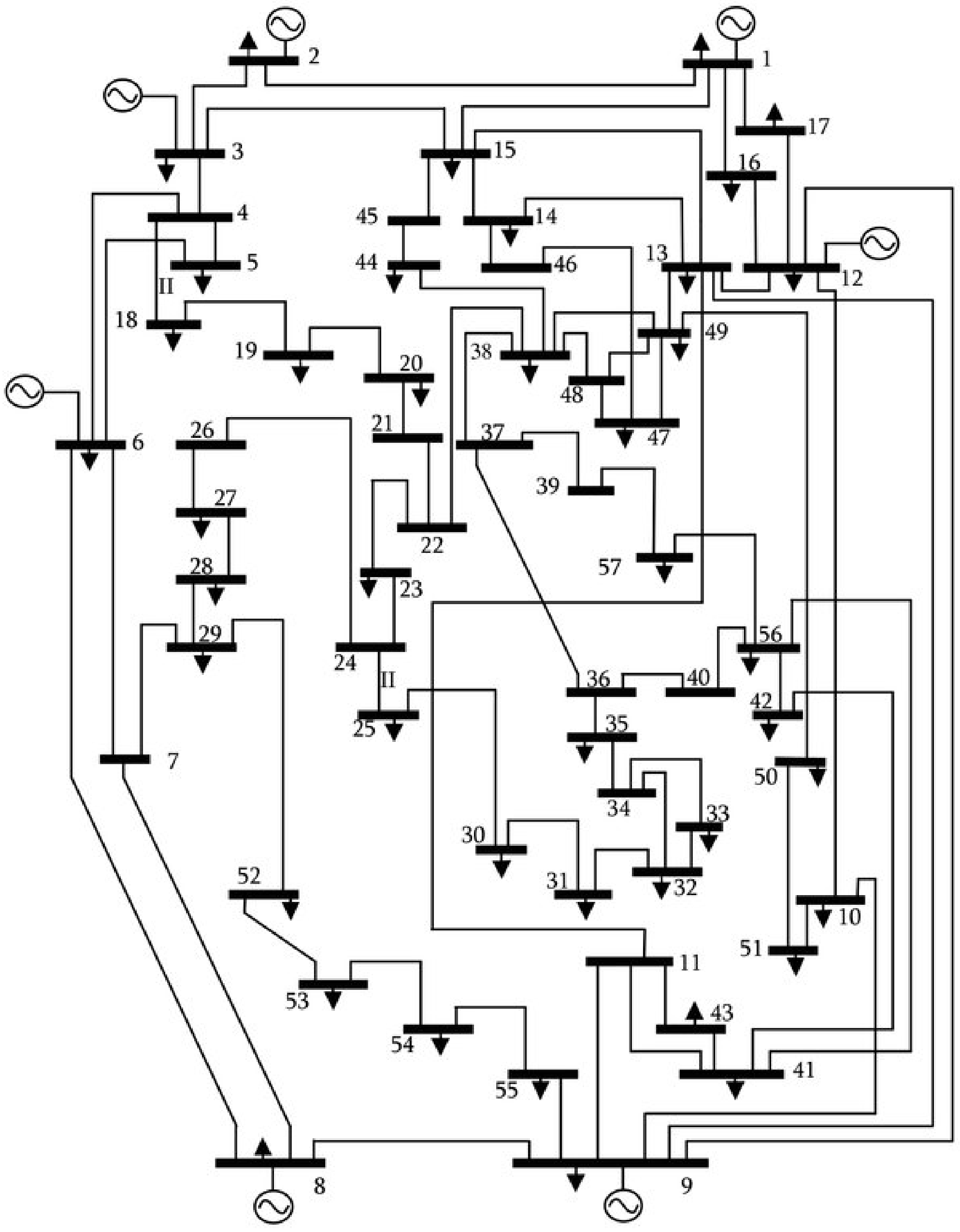


FIGURE 5.4: Diagram of standard IEEE 57-bus system (adapted from [106]).

5.6 DATA ENGINEERING

During the data engineering, also called data wrangling or data preprocessing, phase, data is prepared to be fitted for the subsequent analytics method or tool.

The dataset generated did not have any missing data or invalid data; consequently, no data imputation or cleaning was required. The data types of all the features in the dataset are numeric, except for the class label which is either ‘normal’ or ‘attack’. As part of preprocessing, all the ‘normal’ strings were changed to 0s and ‘attack’ strings to 1s.

5.6.1 *Scaling*

Standard scaling was applied to the dataset. In standard scaling, the features are normalized by removing the mean and scaling to unit variance. Standard scaling replaces the data values in a feature by their z score. For a value x, the z score is calculated as: $z = (x - \mu)/\sigma$, where μ is the mean and σ is the standard deviation of the data values for a given feature.

5.6.2 *Balancing*

The dataset generated contains 90% normal data and 10% attack data implying that the dataset is imbalanced. Supervised classifiers perform poorly when trained with imbalanced datasets, especially for the minority class. In this case, the ‘attacks’ are in the minority class, and the goal is to detect these precisely. Therefore detection of these minor class ‘attack’ instances in an imbalanced dataset will not be accurate. To overcome this problem, two popular techniques to balance datasets, namely the synthetic minority over-sampling technique (SMOTE) and the edited nearest neighbor (ENN), were applied to over-sample the ‘attack’ sets of data and under-sample the ‘normal’ sets of data respectively [13]. After this balancing act the ratio of major and minor class samples in the dataset was 3:2. Since the unsupervised models function as outlier or anomaly detectors, the dataset does not need to be balanced for the classification using unsupervised models.

5.7 FEATURE REDUCTION

The complexity and time for model training increase sharply with an increase in the number of features, i.e., feature dimensionality, in the dataset, because of the so-called “curse of dimensionality” [107]. The number of data items, i.e., features, in SE measurement data increases with the number of buses in the system. For example, the number of features in the measurement data for an IEEE 9-bus system is 27, for an IEEE 14-bus system it is 122, for an IEEE 57-bus system is 547, and for an IEEE 300-bus system it is 1122.

It is often observed that not all the features in a dataset contribute equally in training the models. Hence, the features with the least discriminating properties can be safely eliminated from the dataset without compromising the model performance. Feature selection may lead the trained model to maximizing its performance while minimizing its running time.

5.7.1 *Random Forest Classifier Method*

In the security analytics scheme presented in this dissertation, random forest classifier (RFC) [46] was used to rank and select the features in the SE dataset according to their importance in model training.

Random forest, a popular classification and regression method, can also be used to rank the features in a dataset based on their importance. Random forest is an ensemble of a large number of decision trees. Each of the trees is constructed over a random extraction of the instances from the dataset and a random extraction of the features. These trees are uncorrelated since they can't access all features or all observations. Every node in the trees is a condition on a single feature, designed to split the dataset into two such that similar response values end up in the same set. At the time of this split, a measure of how much each feature contributes in making this decision is taken. This measure forms the basis of ranking the features according to their importance. Then, a number of the most important features are retained from the ordered list of features while the others are deleted from the dataset to obtain a feature-reduced dataset.

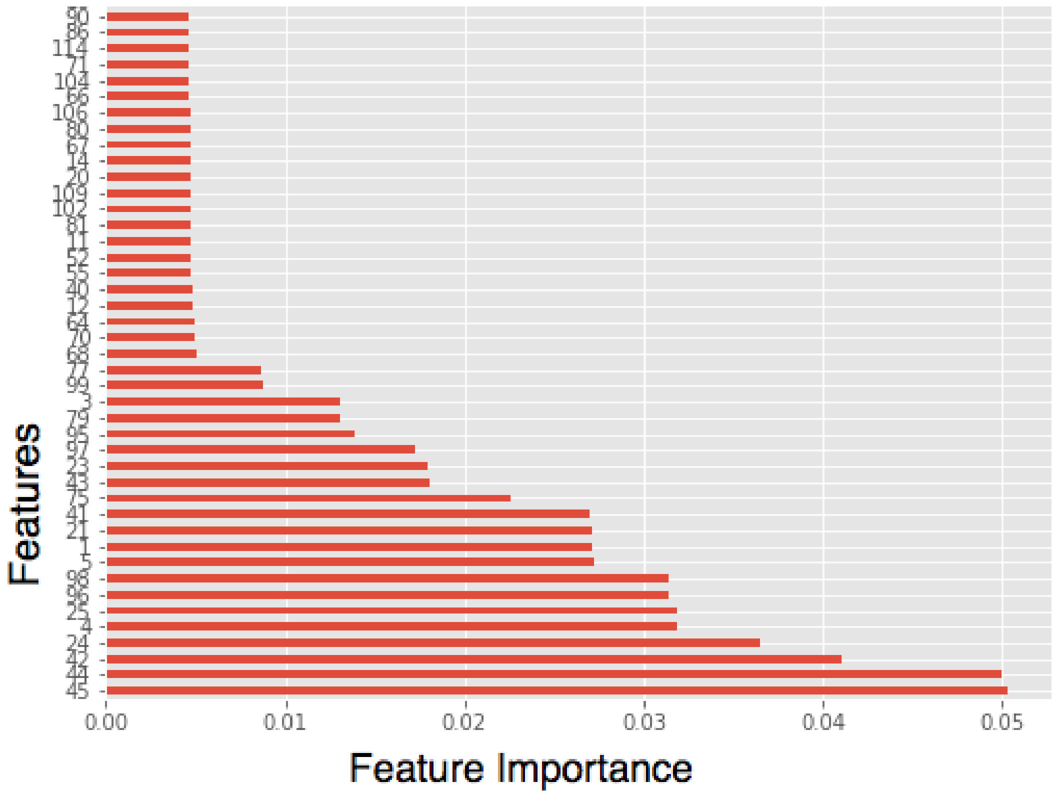


FIGURE 5.5: Features in the 14-bus-AM1 dataset in order of importance.

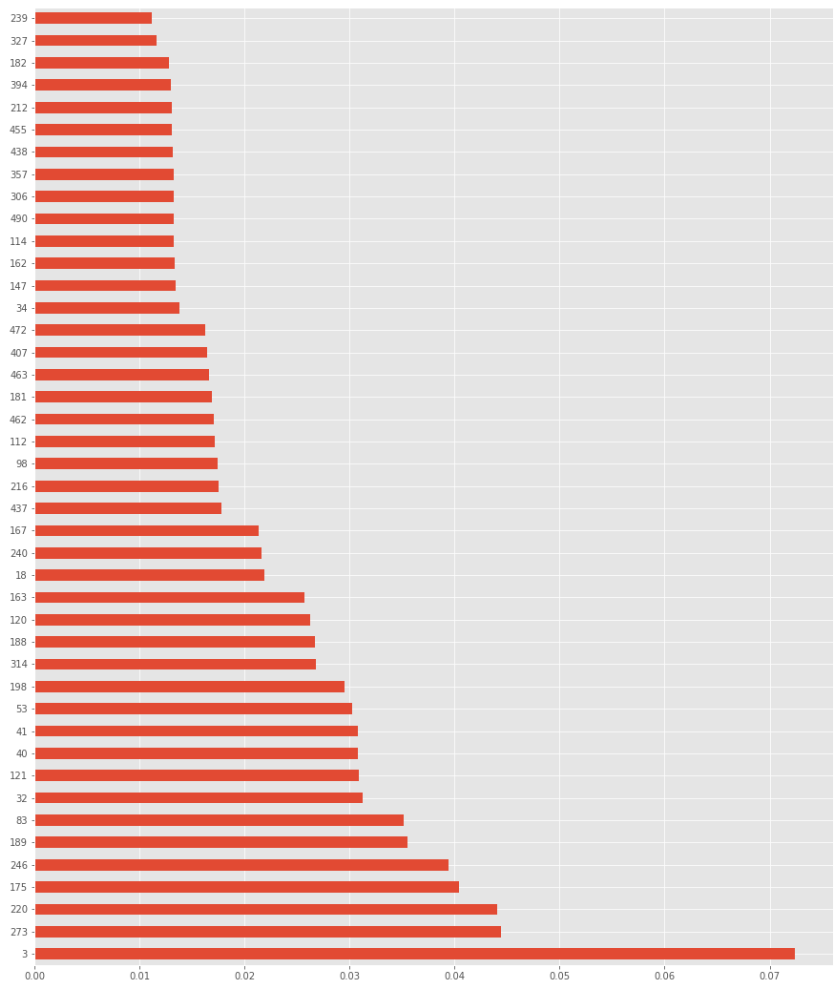


FIGURE 5.6: Features in the 57-bus-AM2 dataset in order of importance.

5.7.2 *Selecting SE Measurement Features*

The random forest classifier was used on the dataset to obtain an ordering of the features according to their importance. A plot showing the feature importance for the 14-bus-AM1 dataset is given in Figure 5.5. The figure shows that the first 21 features have the largest variances, and therefore only these features were retained in the dataset as the predictor variables, plus the target variable. Figure 5.6 shows the feature importance for the 57-bus-AM2 dataset showing that the first 43 features have largest variances, and therefore these features were retained in the dataset.

In summary, the datasets generated in Section 5.5 are scaled and balanced (for supervised methods only) in Section 5.6, and finally feature-reduced in Section 5.7. The properties of the twelve datasets after scaling, balancing, and feature selection are given in Table 5.1. These datasets will be used for model selection, i.e., for training and evaluating machine learning models, in Chapters 6–9.

TABLE 5.1: Properties of the datasets after scaling, balancing, and feature reduction.

Datasets	Total Buses	Buses Attack	Learning Class	Total Data	Normal Data	Attacked Data	Total Features	Selected Features
14-Bus-AM1	14	5	Supervised	112,568	66,980	45,588	122	21
			Unsupervised	100,000	90,124	9,876	122	21
14-Bus-AM2	14	4, 8, 9	Supervised	110,652	66,778	43,874	122	21
			Unsupervised	100,000	90,079	9,921	122	21
14-Bus-AM3	14	All	Supervised	113,567	66,762	46,805	122	21
			Unsupervised	100,000	90,020	9,980	122	21
57-Bus-AM1	57	7	Supervised	106,048	63,515	42,533	547	43
			Unsupervised	100,000	90,034	9,966	547	43
57-Bus-AM2	57	7, 11, 19, 21	Supervised	102,092	61,824	40,268	547	43
			Unsupervised	100,000	90,008	9,992	547	43
57-Bus-AM3	57	All	Supervised	104,582	62,630	41,952	547	43
			Unsupervised	100,000	90,123	9,877	547	43

Table 5.1 shows that the ratio of normal data and attack data, i.e., data tampered with by the attackers, is roughly 3:2 for datasets used in supervised model selections and roughly 9:1 for datasets used in unsupervised model selections. It also shows that the total number of data instances for supervised datasets are more than 100,000. This is because when SMOTEEN method is used to balance a dataset, the method increases the minor class instances and reduces the major class instances. At the end, the number comes something close to 100,000.

5.8 MACHINE LEARNING MODEL SELECTION

A machine learning 'method' is the implementation of a machine learning algorithm. A machine learning 'model' is obtained when the corresponding 'method' is used to train on a particular dataset. It is then said that the 'model' has learned the patterns or relationships in the data instances using the ‘method’. Many models can be built using one method by using different datasets or different sets of parameters for the method. Even if only one dataset and one set of parameters are used, different models can be built using different ways of training, i.e., depending on how the dataset was split or what kind of training was used.

Model selection is the process of training and evaluating machine learning methods and selecting the best model from among a collection of machine learning models for a particular training dataset. The process can be applied to 1) models of different methods (e.g., decision tree, logistic regression, SVM, kNN, etc.), 2) models of the same method configured with different hyper-parameter values (e.g., different hidden layers and units for ANN), and 3) models of the same method and same set of hyper-parameter values trained repeatedly with resampled training data.

In this security analytics framework, all of these three types are used to train the most robust models. For each of the different machine learning methods, a set of hyper-parameter values were chosen and the models were trained using 10-fold cross-validation over randomly divided (resampled) training data. In 10-fold cross validation, the training dataset is divided into 10 equal subsets. The training is repeated 10 times with 9 subsets and the remaining 1 subset, called the validation dataset, is used to test the trained model. In every repetition, a different sets of training subsets and validation subset are used. Figure 5.7 shows the *k*-fold cross-validation process. The performance is calculated by averaging the 10 sets of performance results. This process yields the best model out of several models of one ML method with one set of hyper-parameter values. Then this process is repeated for different sets of hyper-parameter values for that one ML method. The best models obtained from each of these sets of hyper-parameter values are then compared using standard evaluation

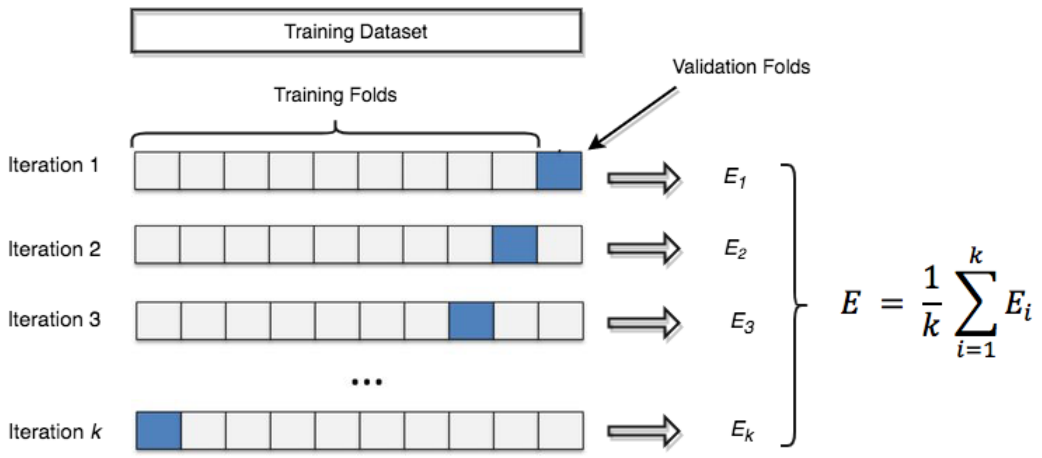


FIGURE 5.7: Diagram showing *k*-fold cross-validation process.

metrics described in Section 5.8.1. Finally, this entire process is applied to all the different methods yielding one best model for each of the different methods. The final best model is then selected by comparing these models using evaluation metrics.

The best models are selected after the models are trained with cross-validation and their performance are measured against the testing dataset using standard evaluation metrics described below. Since the goal is to detect attack data, which is in the minor class, the appropriate metrics to be used to select optimum models are F1-score, sensitivity, and specificity. The ideal situation would be when one model exhibits highest values for all these three metrics. However, that is not always the case. Then, a model is chosen as the best one that gives highest sensitivity along with highest or very close to highest values for F1-score and specificity. These best-performing models are then deployed to detect the attacks in real-time.

Chapters 6–9 will elaborate on this section in describing the model selection with different machine learning methods and will present the results showing how good the models are in detecting SFDI attacks.

5.8.1 *Standard Evaluation Metrics*

A machine learning classification model predicts class labels as output for a given input dataset. In this case, the class labels are “normal” and “attack”. Depending on the predicted class labels, the outcomes for binary classification can be categorized as: 1) True positive (TP): when the model correctly identifies an attack, 2) True negative (TN): when it correctly identifies a normal or non-attack, 3) False positive (FP): when a non-attack is incorrectly identified as an attack, and 4) False negative (FN): when an attack is incorrectly identified as a non-attack. These four categories constitute the so-called *confusion matrix* [42] shown in Table 5.2.

TABLE 5.2: The Confusion Matrix.

	Predicted Normal	Predicted Attack
Actual Normal	TN	FP
Actual Attack	FN	TP

To evaluate the models in this paper, the following metrics derived from the confusion matrix are used in the dissertation [100].

1. Accuracy $= (TP + TN)/TotalPopulation$
2. Precision $= TP/(FP + TP)$
3. False Positive Rate (FPR) $= FP/(FP + TN)$
4. Sensitivity $= TP/(FN + TP)$
5. Specificity $= TN/(FP + TN)$
6. F1-score $= 2TP/(2TN + FP + FN)$

Accuracy is the percentage of correct identification over total data instances. *Precision*, also known as the positive predictive value, represents how often the model correctly identifies an attack. *Sensitivity*, also known as the true positive rate (TPR), recall, or detection rate, indicates how many of the attacks the model does identify correctly. Sensitivity intuitively gives the ability of the classifier to find all the positive samples and the precision intuitively gives the ability of the classifier not to label as positive a sample that is negative. *Specificity*, also known as the true negative rate, measures the proportion of actual negatives, i.e., non-attacks, that are correctly identified as such. *F1-score*, also known as F-measure, provides the harmonic average of precision and sensitivity.

The ROC curve plots FPR on the X-axis and TPR on the Y-axis. This means that the top left corner of the plot is the “ideal” point, where the $FPR = 0$, and $TPR = 1$. The larger the area under the curve (AUC) the better. The red dotted line indicates the random classification and has an AUC of 0.5. The *ROC AUC score* is a measure of the diagnostic ability of binary classifier systems. To demonstrate the detection performance of different models over all possible thresholds, the *ROC curves* are plotted.

In addition, the run times (i.e., elapsed times) are usually measured for comparing the speed of different training models running the all-feature dataset versus the reduced-feature dataset.

5.9 MODEL DEPLOYMENT

One or more best models based on a comparison of the standard evaluation metrics across all models that were tested are then deployed in the system being protected to detect security issues in real-time. For this case, the model(s) are to be deployed at the power control center within the state estimation process and right after the bad-data detection. In this dissertation research, the model deployment phase was not performed. This has been left as one of the future works.

5.10 ATTACK MONITORING

The SE measurement data matrix goes through the steps in data engineering phase, where it is normalized and feature-reduced. Then it goes through the deployed models. If any of the models identifies the presence of a potential SFDI attack then the mechanism raises an alert for the control center operator to take necessary action. Since, the models were not deployed in this research, attack monitoring is also left out as a future work.

5.11 THE SOFTWARE

A software was implemented to perform most of the phases of the security analytics process presented above. This software was used to do data generation, data engineering, feature reduction, model selection, and evaluating the models.

The data generation module was implemented using MATPPOWER [132] and MATLAB, and all the other functionalities were implemented using the Python programming language. Python library scikit-learn [90] was used to write the feature reduction method and machine learning model training and selection.

At this time, the implemented software supports RFC for feature reduction; GLM, GBM, DRF, ANN, SVM, NB, DT, and LR for supervised learning; OCSVM with two kernels, ISOF, LOF, and EE for unsupervised learning; stacking ensembles using the supervised and unsupervised methods; standard and MinMax scaling; evaluation of the model performance and printing out the results in tabular forms; and plotting performance graphs.

5.12 SUMMARY

In this chapter, the security analytics framework used for detecting SFDI attacks on state estimation in power transmission systems is presented. This chapter discusses the steps in the framework and describes what were done for different steps for this research, except the “Machine Learning Model Selection” phase. The following four chapters discuss machine learning model selection using supervised, ensemble, artificial neural networks, and elliptic envelope methods, and demonstrate how good the models will be in detecting SFDI attacks using standard evaluation metrics.

CHAPTER 6

Supervised Models

6.1 Introduction

This is the first of four chapters that represents machine learning model selection phase (Section 5.8) in the security analytics framework. These chapters present building of machine learning models using different machine learning methods using the datasets generated and prepared in Chapter 5. The model selection process ends with determining the efficacy of the models in detecting SFDI attacks using standard evaluation metrics as described in Section 5.8.1.

This chapter describes the model selection using three popular supervised methods, namely generalized linear model (GLM), gradient boosting machine (GBM), and distributed random forests (DRF) and resultant detection strengths of the models.[2]

6.2 Motivation

An empirical approach to identify the best machine learning model(s) to accurately and reliably detect SFDI attacks was started with three popular and simple to use methods. For that reason, a powerful linear method, GLM, a boosting method, GBM, and a bagging method, DRF, are chosen.

[2]The work presented in this chapter has been published in the following paper:
M. Ashrafuzzaman, Y. Chakhchoukh, A. Jillepalli, P. Tosic, D. Conte de Leon, F. Sheldon, and B. Johnson, “Detecting stealthy false data injection attacks in power grids using deep learning,” in IEEE Wireless Communications and Mobile Computing Conference (IWCMC), 14th International, pp. 219–225, IEEE, 2018. doi:10.1109/IWCMC.2018.8450487

6.3 THE SUPERVISED METHODS

This section briefly describes the three machine learning methods used in this chapter.

6.3.1 *Generalized Linear Model*

Generalized linear models (GLM) were first proposed as a general framework for unifying various other statistical models, including linear regression, logistic regression and Poisson regression [86]. A few other linear regression and classification methods have been added to GLMs since then. An implementation of the GLM suite generally includes Gaussian (i.e. normal), Poisson, Gamma and Ordinal regressions as well as binomial and multinomial classifications. The GLM allows the linear model to relate to the response variable using a link function and the magnitude of the variance of each measurement is a function of its predicted value.

6.3.2 *Gradient Boosting Machine*

Gradient boosting machine (GBM) is a forward-learning ensemble method. An ensemble is a collection of weak classifiers. The mean of predictions by all these classifiers is deemed as the final prediction. GBM trains many models in a gradual, additive and sequential manner. GBM overcomes the limitations of the weak classifiers by using gradients in the loss function which is a measure of how good the model's coefficients are at fitting the underlying data.

6.3.3 *Distributed Random Forest*

Distributed random forest (DRF) is another ensemble-based method that works by generating a forest of classification trees, rather than a single classification or regression tree, using the given dataset. Each of the trees in the forest is a weak classifier built on a subset of rows and columns. More trees will obviously reduce the variance. The average prediction over all of these trees is used to make a final prediction.

6.4 MODEL SELECTION

Each of the six datasets was split into a 7:3 ratio for training and testing dataset respectively. The training datasets are used to train the three models and the testing datasets are used to evaluate the models for selection using the standard metrics. 10-fold cross-validation over randomly divided training data was used during training of the models.

6.5 RESULTS

The numerical results in terms of the evaluation metrics for testing the models using test datasets are shown in Table 6.1.

TABLE 6.1: Evaluation metrics values for the three supervised methods for all the six datasets.

Datasets	**Model**	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**ROC AUC**	**Elapsed Time**
14-Bus-AM1	GLM	0.8343	0.8728	0.9821	0.7234	0.9827	0.8522	15.55s
	GLB	0.8356	0.8736	0.9843	0.7245	0.9816	0.8576	16.95s
	DRF	0.8349	0.8787	0.9844	0.7258	0.9832	0.8577	21.23s
14-Bus-AM2	GLM	0.8343	0.8727	0.9819	0.7232	0.9826	0.8521	14.43s
	GLB	0.8355	0.8734	0.9842	0.7242	0.9815	0.8573	15.88s
	DRF	0.8350	0.8785	0.9841	0.7254	0.9833	0.8574	21.34s
14-Bus-AM3	GLM	0.8342	0.8728	0.9820	0.7234	0.9826	0.8519	15.21s
	GLB	0.8357	0.8737	0.9844	0.7243	0.9817	0.8577	16.45s
	DRF	0.8348	0.8786	0.9842	0.7259	0.9833	0.8576	21.02s
57-Bus-AM1	GLM	0.8334	0.8719	0.9814	0.7222	0.9821	0.8511	17.35s
	GLB	0.8347	0.8726	0.9832	0.7234	0.9816	0.8559	18.78s
	DRF	0.8340	0.8777	0.9836	0.7246	0.9828	0.8562	23.12s
57-Bus-AM2	GLM	0.8335	0.8721	0.9816	0.7224	0.9823	0.8512	17.47s
	GLB	0.8349	0.8728	0.9837	0.7233	0.9818	0.8561	18.66s
	DRF	0.8344	0.8777	0.9839	0.7248	0.9829	0.8563	23.35s
57-Bus-AM3	GLM	0.8334	0.8722	0.9818	0.7225	0.9822	0.8516	18.02s
	GLB	0.8347	0.8735	0.9838	0.7238	0.9817	0.8558	18.95s
	DRF	0.8340	0.8779	0.9839	0.7249	0.9827	0.8564	23.43s

6.6 DISCUSSION OF RESULTS

The following observations are made from Table 6.1 for the evaluation metrics values for the three ML models for the six datasets:

1. The performance for all the three ML models are effectively same.
2. The performance of these models for all the six datasets are also effectively the same.
3. The overall accuracy values for the models are 87%, whereas precision values are about 98%. However, in a classification problem where the goal is to detect the minor class occurrences, the most important metrics are the sensitivity which, in this case, measures the proportion of actual "attacks" that are correctly identified as such; and the specificity which measures the proportion of actual "non-attacks" that are correctly identified as such. For these models, the sensitivity values are about 72%. This indicates that even the any of these model would be able to detect about 72% of the attacks and the rest 28% will go undetected. The specificity values for the models are about 98% meaning that the models are able to identify a "non-attack" as such 98 times out of 100, with 2 as false alarms.

The corresponding full-feature datasets (122 features for 14-bus system and 547 features for 57-bus system) were also used to train the three models. It was found that the performance numbers for the models do not change at all for these cases. However, it was found that the elapsed times for training with the full-feature datasets for 14-bus system take up to 400% more time, and for 57-bus system take up to 600% more time.

6.7 SUMMARY

In this chapter, use of the security analytics framework with supervised learning in the model selection phase was described for detection of SFDI attacks. The supervised models used are GLM, GBM and DRF. It cannot be confidently said that the models performed well in detecting the attacks as the average detection rate obtained by these models is only around 72%. It was found that using the datasets with all the features do not improve the performance values, however it slows the training down significantly.

Chapter 7
Stacking Ensembles

7.1 Introduction

This chapter presents the model selection part of the security analytics framework using stacking ensemble. It also describes the ensemble framework[3] and building of machine learning models using machine learning methods in the ensemble framework using the datasets generated and prepared in Chapter 5. The model selection process ends with determining the efficacy of the ensemble models in detecting SFDI attacks using standard evaluation metrics as described in Section 5.8.1.

The ensemble framework employs two sets of machine learning stacking ensembles. One set of the ensembles uses only supervised methods while the other one uses only unsupervised methods. In addition to the ensemble classifiers, the performance of the individual classifiers are evaluated for comparison.[4]

7.2 Motivation

Different classifiers usually perform differently on the same data. In stacking ensemble-based machine learning, multiple different classifiers are used together and the results given by these constituent classifiers are further classified by another (second stage)

[3]The ensemble framework is a part of the security analytics framework.

[4]The work presented in this chapter has been published in the following two papers:

1. M. Ashrafuzzaman, S. Das, Y. Chakhchoukh, S. Shiva, and F. T. Sheldon. “Detecting stealthy false data injection attacks in the smart grid using ensemble-based machine learning,” Journal of Computers & Security, Elsevier, August 2020. doi:10.1016/j.cose.2020.101994
2. M. Ashrafuzzaman, S. Das, Y. Chakhchoukh, S. Duraibi, S. Shiva, and F. T. Sheldon. “Supervised learning for detecting stealthy false data injection attacks in the smart grid,” Transactions on Computational Science and Computational Intelligence, Advances in Security, Networks, and Internet of Things, Springer Nature, 2020. (accepted)

classifier [28, 91]. This approach is called a two-layer ensemble or stacking [114]. Stacking ensemble-based machine learning approaches have been shown to perform well in solving other problems [26, 83, 129].

In supervised learning, labeled data, i.e., a training set of examples with correct responses, is provided and based on this training the machine learning method generalizes (i.e., learns the patterns in the training data) to classify the unlabeled input sets. In unsupervised learning, the method is trained with unlabeled data to identify similarities between the inputs that have something in common. These similar inputs are then categorized together. In other words, the method attempts to learn the hidden patterns in the input data, and later predicts responses to test inputs based on the learned patterns.

If trained with well-developed labeled data, supervised learning models perform better than unsupervised models. However, the additional and often cumbersome data engineering needed to label raw data is painstaking and challenging. Moreover, attack data is very sparse. Consequently, the availability of labeled data is not always guaranteed in a timely manner. That is why most datasets are synthetic. Because supervised models are trained with labeled data, they learn the patterns associated with "attacks" very well and can detect such patterns more consistently. However, if the attack pattern is not one of the learned patterns or if the attack is a new one, then the performance of supervised models is highly degraded. In these cases, unsupervised models which flag any out-of-the-ordinary pattern more consistently must be considered. For these reasons, both supervised and unsupervised models are included in the ensemble framework.

7.3 METHODS USED IN ENSEMBLE

This section briefly describes the machine learning methods used in the ensemble framework. More information on these learning methods can be found in the books by Hastie et al. [42] and Marsland [76].

7.3.1 *Supervised Learning Algorithms*

7.3.1.1 *Logistic Regression*

Logistic regression (LR) is a classification algorithm that performs very well on linearly separable classes. LR builds on ideas from the field of statistics where the logistic model is used to discern the probability of a true/false class or event. This can be extended to models with more than two classes of events. Each of the events would be assigned a probability value between 0 and 1, where the sum of all probabilities is unity. The coefficients of the logistic regression algorithm must be estimated from the training data which is done by using maximum-likelihood estimation. The best coefficients would result in a model that would predict a value very close to 1 for the default class and value very close to 0 for the other class.

7.3.1.2 *Support Vector Machine*

Support vector machine (SVM) is a group of supervised learning methods that identify the patterns for data classification or regression analysis based on finding a separating hyperplane in the feature space between two classes, in such a way that the distance between the hyperplane and the closest data points for each class is maximized. The SVM algorithm is based on probabilistic statistical learning theory [24] whereby the approach favors a minimized classification risk rather than optimal classification. Various types of dividing classification surfaces can be realized by applying a kernel, such as linear, polynomial, Gaussian, radial basis function (RBF), sigmoid, or hyperbolic tangent [23].

7.3.1.3 *Naïve Bayes*

Naïve Bayes is a simple classifier used in many machine learning problems. Based on the Bayes theorem this probabilistic classifier helps define the probability of an event based on some prior knowledge of certain conditions associated with that event. The name naïve Bayes originates from the fact that the input features are assumed to be independent, even though in practice this may not always be true.

7.3.1.4 *Decision Tree*

Decision tree (DT) is a non-parametric supervised learning method and is used both for classification and regression. DT builds a tree in which each branch shows a probability between a number of possibilities and each leaf shows a decision. The paths from the root of the tree to the leaves represent classification rules. The algorithm collects information and applies the rules for the purpose of the decision to take a particular path. In DT, each level splits the data according to different attributes and attempts to achieve perfect classification with a minimal number of decisions.

7.3.1.5 *Artificial Neural Networks*

The computational architecture of neural networks mimics the neural structure and function of the brain forming the interconnected groups of artificial neurons. Each of these artificial neurons is a set of input values and associated weights that trigger the neuron beyond a threshold. The neural network is organized in layers of neurons. The first layer is the input layer and the last one is the output layer. The layers in between these two are called hidden layers. The neural networks attempt to hierarchically learn deep features and correlations in input data by adjusting the weight associated with the neurons. Neural network architectures have many variants with each finding success in specific domains of applications. An extensive review of neural networks can be found in the paper by Schmidhuber [98].

7.3.2 *Unsupervised Learning Algorithms*

7.3.2.1 *One-Class SVM*

One-class classification, also known as unary classification or class-modeling, tries to identify objects of a specific class primarily by learning from an unlabeled training set containing only the objects of that class. Trained in this way, the classifier flags any object not recognized according to the learned generalization as an outlier or anomaly [53]. OCSVM [99] is a support vector machine based anomaly detector. Like the supervised SVM models, unsupervised one-class versions also work with different kernels. In the ensemble framework, one-class SVMs with a linear kernel and a polynomial kernel are used.

7.3.2.2 *Isolation Forest*

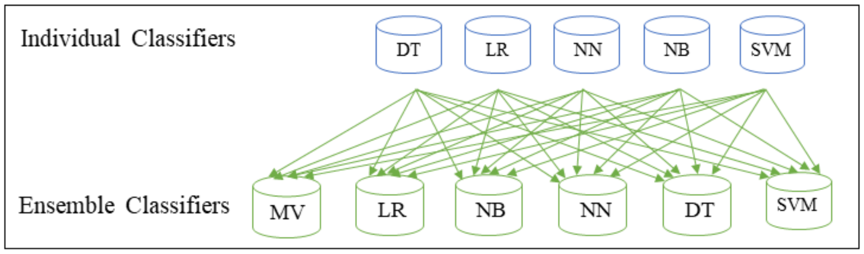
Isolation forest [65] is an ensemble regressor consisting of a number of isolation trees. Each tree is trained on a random subset of the training data. Isolation forest is used to perform the outlier detection efficiently in high-dimensional datasets. The algorithm isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected features using recursive partitioning. The required number of splits to isolate a sample is equivalent to the path length from the root to the leaf. The path length from the root to leaf, averaged over a forest of such random trees, is a measure of normality and the decision function. Random partitioning produces noticeably shorter paths for anomalies. Therefore, the trees that collectively produce shorter paths for particular samples are highly likely to be anomalies.

7.3.2.3 *Elliptic Envelope*

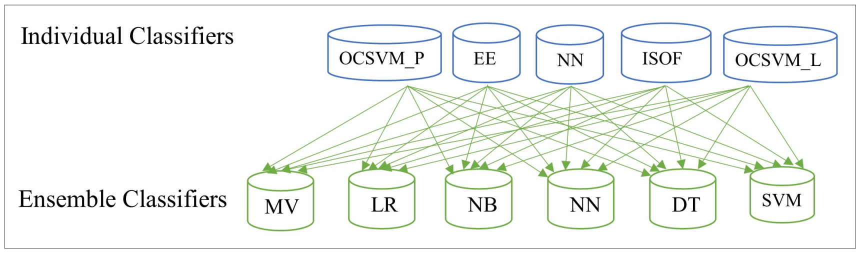
Elliptic envelope (EE) attempts to 'draw' an ellipse putting the normal class members inside the ellipse. Any observation outside the ellipse is then classified as an outlier or anomaly. EE models the data as a high dimensional Gaussian distribution with possible covariances between feature dimensions and uses the FAST-Minimum Covariance Determinant [47, 97] to estimate the size and shape of the ellipse.

7.3.2.4 *Local Outlier Factor*

The local outlier factor (LOF) [15] first computes the local densities of the data objects using distances given by k-nearest neighbors (k-NN) algorithm. LOF then identifies regions of similar density by comparing the local density of a given data object to the local densities of its neighbors. The data objects that have substantially lower densities than their neighbors are identified as the anomalies.



(a) Supervised Ensemble Models.



(b) Unsupervised Ensemble Models.

FIGURE 7.1: Five individual classifiers combining with one of six ensemble classifiers to form six ensembles.

7.4 ENSEMBLE-BASED FRAMEWORK

This section provides an overview of the stacking ensemble-based SFDI attack detection scheme. The ensemble framework (a smaller framework within the overall security analytics frame) consists of both supervised and unsupervised classifiers.

7.4.1 *Individual Classifiers*

In this ensemble-based approach, two different ensembles are used: one using five supervised classifiers and another one five unsupervised classifiers. These individual classifiers when run together, simultaneously or one after another, form the first phase of the stacking ensemble.

7.4.1.1 *Supervised Classifiers*

The supervised classifiers included in the ensemble framework are: decision tree (DT), logistic regression (LR), naïve Bayes (NB), support vector machine (SVM), and artificial neural network (ANN).

7.4.1.2 *Unsupervised Classifiers*

The unsupervised classifier models included in the ensemble framework are: one-class support vector machine with polynomial kernel (OCSVM_P), one-class support vector machine with linear kernel (OCSVM_L), isolation forest (ISOF), elliptic envelope (EE), and local outlier factor (LOF).

7.4.2 *Stacking Ensemble Classifiers*

Stacking ensemble is a learning method where different classifiers are combined into a meta-classifier that has better generalization performance than each individual classifier alone [28, 91]. Stacking ensemble is a two-stage process. The first stage consists of different classification methods, for example, DT, SVM, LR, and so on, or one base classification algorithm can be used repeatedly with different subsets of the training data. These individual classifiers are run together, simultaneously or one after another. In the second stage, the decisions given by individual classifiers are fed as input to another classifier, called the *ensemble classifier*, for the final decision. Majority voting is the most popular ensemble method, which selects the class label that has been predicted by the majority of the constituent classifiers. Instead of majority voting, any binary classification method can be used as the ensemble classifier. The ensemble framework uses six classifiers as the ensemble classifier: majority voting (Ens_MV), logistic regression (Ens_LR), naïve Bayes (Ens_NB), neural network (Ens_NN), decision tree (Ens_DT), and support vector machine (Ens_SVM).

7.5 MODEL SELECTION

This section presents a set of model selection experiments that were performed using the ensemble framework presented above. The goal of the experiments is to find one or more machine learning models in the ensemble framework that can detect SFDI attacks accurately and reliably. The experiments were conducted with individual classification first and then ensemble classification. The experiments ran the data through five supervised and five unsupervised learning models. Six ensemble models were run with the outcomes of the supervised models, and six ensemble models with the outcomes of the unsupervised models. The hyper-parameters used for different models are given in Table 7.1. The dataset was split into two subsets: 70% for training and 30% for testing. To avoid over-fitting and to obtain robust models, 10-fold cross validation over randomly divided training data was used. Then the test data was used for prediction and for measuring model performance.

TABLE 7.1: Hyper-parameter values used for different individual and ensemble classifiers.

	Classifier	**Short Names**	**Hyper-parameter Values**
Supervised Models	Logistic Regression	LR	default parameters
	Decision Tree	DT	default parameters
	Naïve Bayes	NB	alpha=1.0, binarize=0.0, fit_prior=True, class_prior=None
	Neural Network	NN	solver='adam', alpha=1e-5, hidden_layer_sizes=(20, 20), random_state=1
	Support Vector Machine	SVM	C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=True
Unsupervised Models	One Class SVM-Polynomial Kernel	OCSVM_P	nu=0.2, kernel='poly', gamma=0.1
	One Class SVM-Linear Kernel	OCSVM_L	nu=0.2, kernel='linear', gamma=0.1
	Local Outlier Factor	LOF	n_neighbors=20, contamination=0.1, novelty=True
	Isolation Forest	ISOF	behaviour='new', max_samples=100, random_state= rng, contamination=0.1
	Elliptic Envelope	EE	support_fraction=1, contamination=0.03, random_state = rng
Ensemble Models	Majority Voting	Ens_MV	none
	Decision Tree	Ens_DT	default parameters
	Naïve Bayes	Ens-NB	alpha=1.0, binarize=0.0, fit_prior=True, class_prior=None
	Logistic Regresion	Ens_LR	random_state=0, solver='lbfgs', multi_class='multinomial'
	Neural Network	Ens_NN	solver='adam', alpha=1e-5, novelty=True hidden_layer_sizes=(5), random_state=1
	Support Vector Machine	Ens_SVM	C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=True

7.6 RESULTS

Table 7.4 lists representative individual and ensemble models for different datasets and their corresponding evaluation metrics values. Detail results of all the individual and ensemble models using different datasets and the associated evaluation metrics values are given in Appendix A.1.

TABLE 7.2: Training times for supervised individual and ensemble models for the 14-bus-AM1 dataset.

		Elapsed Time	
Models	**21 Features**	**122 Features**	**Without SVM**
LR	0.56s	1.02s	
NB	0.27s	1.03s	
NN	0.57s	0.83s	
DT	1.59s	1m 55s	
SVM	45m 14s	2h 28m	
Ens_MV	45m 19s	2h 30m	6.07s
Ens_LR	45m 17s	2h 30m	4.15s
En_NB	45m 17s	2h 30m	4.12s
Ens_NN	45m 17s	2h 30m	4.33s
Ens_DT	45m 17s	2h 30m	4.08s
Ens_SVM	45m 33s	2h 31m	8.55s

TABLE 7.3: Training times for unsupervised individual and ensemble models for the 14-bus-AM1 dataset.

		Elapsed Time	
Models	**21 Features**	**122 Features**	**Without LOF**
OCSVM_L	35.14s	3m 20s	
OCSVM_P	4m 44s	2h 13m	
ISOF	7.87s	19.16s	
EE	12.39s	1m 24s	
LOF	14m 1s	50m 48s	
Ens_MV	20m 13s	3h 9m	6m 12s
Ens_LR	20m 11s	3h 9m	6m 10s
En_NB	20m 11s	3h 9m	6m 10s
Ens_NN	20m 11s	3h 9m	6m 10s
Ens_DT	20m 11s	3h 9m	6m 10s
Ens_SVM	20m 24s	3h 9m	6m 23s

TABLE 7.4: Representative supervised and unsupervised, individual and ensemble models for the six datasets.

Dataset	Learning Type	Classifier Category	Models	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC	Elapsed Time
14-Bus-AM1	Supervised	Individual	All Models	0.8439	0.8931	0.9991	0.7304	0.9997	0.8650	0.57s
		Ensemble	All Models	0.8472	0.8961	0.9993	0.7353	0.9997	0.8675	45m 17s
	Unsupervised	Individual	EE	0.6318	0.9214	0.5606	0.7237	0.9418	0.8327	12.39s
		Ensemble	Ens_NN	0.6218	0.9216	0.5428	0.7278	0.9505	0.8341	20m 11s
14-Bus-AM2	Supervised	Individual	All Models	0.8472	0.8913	0.9989	0.7354	0.9987	0.8951	0.59s
		Ensemble	All Models	0.8477	0.8911	0.9991	0.7361	0.9989	0.8686	45m 17s
	Unsupervised	Individual	EE	0.6534	0.9379	0.7031	0.6107	0.9728	0.7923	12.4s
		Ensemble	Ens_SVM	0.6521	0.9321	0.7083	0.6048	0.9701	0.7893	20m 11s
14-Bus-AM3	Supervised	Individual	All Models	0.8479	0.8974	0.9994	0.7363	0.9998	0.8954	0.62s
		Ensemble	All Models	0.8482	0.8964	0.9996	0.7367	0.9999	0.8682	45m 18s
	Unsupervised	Individual	EE	0.6540	0.9392	0.7034	0.6110	0.9733	0.7921	12.37s
		Ensemble	Ens_NB	0.6529	0.9395	0.7089	0.6051	0.9743	0.7896	20m 11s
57-Bus-AM1	Supervised	Individual	All Models	0.8369	0.8826	0.9897	0.7288	0.9921	0.8591	0.60s
		Ensemble	All Models	0.8424	0.8893	0.9903	0.7261	0.9921	0.8591	45m 18s
	Unsupervised	Individual	EE	0.6331	0.9189	0.5564	0.7198	0.9389	0.8327	12.4s
		Ensemble	Ens_NN	0.6202	0.9188	0.5375	0.7201	0.9486	0.8341	20m 12s
57-Bus-AM2	Supervised	Individual	All Models	0.8449	0.8862	0.9991	0.7305	0.9994	0.9056	0.62s
		Ensemble	All Models	0.8449	0.8862	0.9991	0.7305	0.9994	0.8652	45m 18s
	Unsupervised	Individual	EE	0.6420	0.9388	0.7115	0.5841	0.9751	0.7803	12.39s
		Ensemble	Ens_SVM	0.6457	0.9402	0.7353	0.5757	0.9788	0.7771	20m 11s
57-Bus-AM3	Supervised	Individual	All Models	0.8453	0.8866	0.9996	0.7322	0.9998	0.9059	0.61s
		Ensemble	All Models	0.8453	0.8866	0.9996	0.7322	0.9998	0.8655	45m 19s
	Unsupervised	Individual	EE	0.6421	0.9390	0.7119	0.5848	0.9756	0.7802	12.38s
		Ensemble	Ens_SVM	0.6462	0.9402	0.7359	0.5760	0.9791	0.7772	20m 10s

7.7 DISCUSSION OF RESULTS

The following observations are made from the tables above:

1. For a particular dataset, the performance values for all the supervised individual classifiers and all the supervised ensemble classifiers are effectively the same for all the metrics.

2. Moreover, the performance values for all the supervised individual classifiers and all the supervised ensemble classifiers are effectively the same for all the metrics for all six datasets, i.e., the performance numbers do not vary for different datasets.

3. The supervised ensemble models do not exhibit better performance than the supervised individual models.

4. The precision values for the supervised models are very close to 100%, whereas accuracy values are about 90%. However, in a classification problem where the goal is to detect the minor class occurrences, the most important metrics are the sensitivity which, in this case, measures the proportion of actual "attacks" that are correctly identified as such; and the specificity which measures the proportion of actual "non-attacks" that are correctly identified as such. For supervised models, the sensitivity values for all the models are very similar, varying between 73.53%. This indicates that the models would be able to detect at most 73% of the attacks and the rest 27% will go undetected. The specificity values for the models are 99.97% meaning that the models are able to identify a "non-attack" as such almost always, and will seldom raise a false alert.

5. For unsupervised models, the performance for different models vary widely for any dataset.

6. For six different datasets also the performance of unsupervised models vary.

7. The performance numbers of the best-performing unsupervised individual and ensemble models are very close to each other for any particular dataset.

8. Out of six datasets, the best-performing ensemble models are a bit better than corresponding best-performing individual models. Therefore, it cannot be concluded either way that ensembles are better than individuals or vice versa.

9. Among the unsupervised individual models, EE performed the best for all six datasets.

10. Among the unsupervised models, the best sensitivity of 72.78% is given by Ens_NN for 14-Bus-AM1 dataset. The corresponding specificity value is 95.05%.

11. Among the six datasets, the lowest sensitivity of 58.41% is given by EE. The corresponding specificity value is 97.51%.

Table 7.2 shows the elapsed time for training the different supervised models. It is notable that not only the ensemble models do not perform any better, but they also take more time to run than the individual models. This is because the ensembles first run all the five individual models and then run the ensemble classifier, and the accumulated elapsed time is therefore higher. The same can be said about the unsupervised models, as shown in Table 7.3.

As seen in Chapter 2, the SVM is a popular model among the researchers working on the problem of detecting stealthy FDI attacks on SE. However, the experiment shows that SVM performs the same as the other models. Moreover, SVM takes much more time to train. Whereas the other individual models take less than 2 seconds to train, SVM takes more than 2700 seconds or 45 minutes on the feature-reduced dataset. On the original dataset with 122 features, SVM takes an astounding 8900 seconds or 2.47 hours. This also exemplifies the “curse of dimensionality” and how it can be handled by reducing the feature set. It was also observed that if SVM is taken out as a constituent individual model, then the times taken by the ensemble models were reduced drastically without any reduction in performance. The last columns in Tables 7.2 and 7.3 show times taken by the ensemble models when SVM or LOF models are not included in the set of the individual models.

The experiment also used the corresponding full-feature dataset (122 features for 14-bus system) to train the models. It was found that the performance numbers for the models do not change at all for these cases. However, it was found that the elapsed times for training with the full-feature datasets for 14-bus system take over 300% more time. For unsupervised models, the elapsed times are 900% higher respectively for 14-Bus-AM1 dataset.

7.8 SUMMARY

In this chapter, use of the security analytics framework with ensemble learning in the model selection phase was described for detection of SFDI attacks. The performance of the ensemble models were compared with the performance of individual models. It was found that the performance of individual supervised models are the same as those of the ensemble models. However, for the unsupervised models, the ensemble models performed better for two datasets, and the individual models performed better for the other four datasets. The best detection rate found for any of the models is 73%. For both the supervised and unsupervised models, reducing the feature set increases the training speeds by many folds without suffering any degradation in detection rates.

CHAPTER 8

Artificial Neural Networks

8.1 Introduction

In this chapter, the model selection part of the security analytics framework using artificial neural networks (ANN) is presented. The ANN models are built using ANN methods training on the datasets generated and prepared in Chapter 5. The model selection process ends with determining the efficacy of the ANN models in detecting SFDI attacks using standard evaluation metrics as described in Section 5.8.1.

8.2 Motivation

So far in the previous two chapters, eight supervised models, five unsupervised models, and six stacking ensemble models were used to detect SFDI attacks with decent detection rate. However, none of the models showed any acceptable performance. Moreover, all the supervised models returned effectively the same detection rate of about 73%, with artificial neural network showing a little better results than the others. Therefore, as a next logical step, a decision to treat artificial neural networks through full range of model selection was taken because ANNs have been known to learn complicated structures in the data instances.

8.3 Artificial Neural Network Method

The artificial neural network, also known as feedforward neural networks (FFNN) or multilayer perceptrons (MLPs), are the quintessential first-generation deep learning models. The main idea for deep learning came from Hecht-Nielsen who proved the universal expressive power of a three-layer neural network back in 1989 [44]. However, it needed the development of a training algorithm by Hinton in 2006 to make way for harnessing the power of this model and for practical implementation architectures [45].

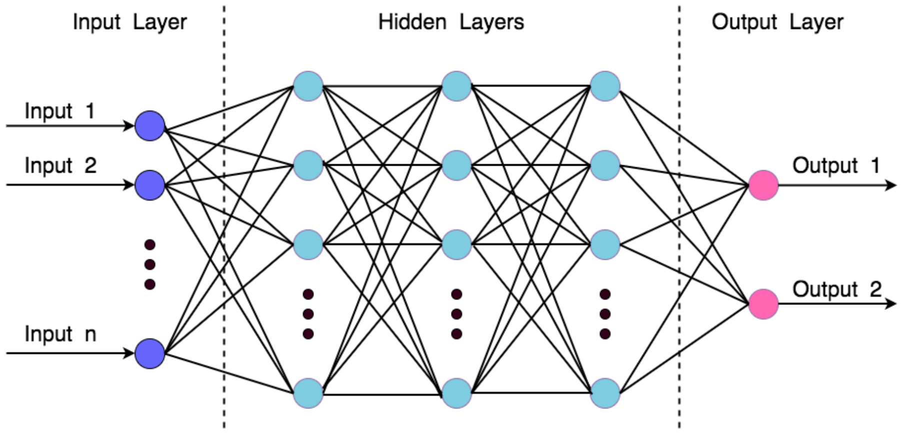


FIGURE 8.1: Basic architecture of an artificial neural network showing the input layer (L1), two hidden layers (L2 and L3) and the output layer (L4) with two responses.

The computational architecture of neural networks mimics the neural structure and function of the brain forming the interconnected groups of artificial neurons. Each of these artificial neurons is a set of input values and associated weights that trigger the neuron beyond a threshold. The neural network is organized in layers of neurons. The first layer is the input layer and the last one is the output layer. The layers in between these two are called hidden layers. The basic architecture has an input layer of nodes equal to the number of input data, one or more hidden layers with varying number of nodes or neurons, and an output layer with number of nodes equal to the number of responses. Figure 8.1 shows a basic architecture of an artificial neural network with the input layer, 2 hidden layers and the output layer. The neural networks attempt to hierarchically learn deep features and correlations in input data by adjusting the weight associated with the neurons. Neural network architectures have many variants with each finding success in specific domains of applications. An extensive review of neural networks can be found in the paper by Schmidhuber [98].

8.4 MODEL SELECTION

This section presents the model selection process with artificial neural networks. The goal of the process is to identify an ANN model that can detect SFDI attacks accurately and reliably. To find out the best-performing ANN model, the ANN method was run through a grid-search with hidden layers and hidden units as the varying hyper-parameters. The hidden layers were chosen from 2, 4, and 6; and with each of these hidden layers, hidden units were chosen between 50 and 2500. Other hyper-parameters used are given below.

```
batch_size='auto',
beta_1=0.9,
activation= 'relu',
solver='adam',
beta_2=0.999,
early_stopping=False,
epsilon=1e-08,
alpha=0.0001,
learning_rate_init=0.001,
max_iter=500,
momentum=0.9,
power_t=0.5,

random_state=None,

n_iter_no_change=10,

shuffle=True,

tol=0.0001,

validation_fraction=0.1,

warm_start=False.
```

TABLE 8.1: Evaluation metrics values for the best-performing ANN models for the six datasets.

Datasets	Hidden Layers	Hidden Units	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC	Elapsed Time
14-Bus-AM1	4	1200	0.9834	0.9853	0.9744	0.9824	0.9878	0.9956	23h 50m
14-Bus-AM2	4	1200	0.9832	0.9855	0.9747	0.9824	0.9878	0.9956	22h 12m
14-Bus-AM3	4	1200	0.9836	0.9855	0.9746	0.9826	0.9887	0.9956	23h 56m
57-Bus-AM1	4	1200	0.9827	0.9847	0.9739	0.9819	0.9872	0.9951	1d 9h
57-Bus-AM2	4	1200	0.9829	0.9850	0.9740	0.9820	0.9874	0.9954	1d 3h
57-Bus-AM3	4	1200	0.9831	0.9853	0.9742	0.9822	0.9876	0.9956	1d 6h

8.5 RESULTS

Table 8.1 lists the best ANN models for different datasets and their corresponding evaluation metrics values. Detail results of ANN model selection using different datasets and associated evaluation metrics values are given in Appendix A.2.

8.6 DISCUSSION OF RESULTS

From Table 8.1 above, the followings are observed:

1. For all the six datasets, the best-performing ANN models are the ones with 4 hidden layers and 1200 hidden units in each of the hidden layers. These

particular models were chosen because these models have highest sensitivity, specificity, and F1-score values.

2. The sensitivity values for the best-performing models are around 98.25%, meaning the models can correctly detect over 98% of the attacks. The specificity values are around 98.75%, meaning that on less than 1.25% times the models would raise a false alarm. These results are by far the better results that were seen with other models in this research. As a matter of fact, these results are among the best results for SFDI attack detection in the literature.

3. Models with more than 4 layers and 1200 units were also trained. However, it was observed that the performance ratings actually go down with higher layers and units after reaching the peaks at 4x1200.

4. The time taken to train the best models are around a day and a half, i.e., about 36 hours. The training was done on a iMac desktop with an Intel®Core™i5 CPU@2.7GHz and 20GB memory running a 64-bit High Sierra macOS. This is not a high-performance machine for this kind of heavy-duty computing. Moreover, the machine was not dedicated to this task. The training time will reduce drastically when a high-performance computing machine with graphical processing units (GPU) is used. Moreover in practice, the training is seldom done in real-time. Instead, the training is used off-line to obtain a robust model and then this "trained" model is deployed online to detect real-time attacks.

8.7 SUMMARY

In this chapter, use of the security analytics framework with artificial neural networks in the model selection phase was described for detection of SFDI attacks. Model selection was performed using artificial neural network models with wide range of hyper-meter settings. It was found that models with 4 hidden layers and 1200 hidden units in each hidden layers can detect 98.25% attacks while raising only 1.25% false alarms. These results are the best it was seen in this research and are among the best in the literature.

CHAPTER 9

Elliptic Envelope

9.1 INTRODUCTION

In this chapter, the model selection part of the security analytics framework using elliptic envelope (EE) method is presented. The EE models are built using EE methods with different hyper-parameter values and training on the datasets generated and prepared in Chapter 5. The model selection process ends with determining the efficacy of the EE models in detecting SFDI attacks using standard evaluation metrics as described in Section 5.8.1.[5]

9.2 MOTIVATION

Supervised methods have the advantage of learning to differentiate the anomalous from normal data using a tagged or labeled dataset. Unsupervised methods, on the other hand, sort data into different clusters where the strength of the clustering lies within the algorithm itself. Among unsupervised methods, novelty and outlier detection strategies have shown significant promise in detecting otherwise hidden anomalies [58, 26]. The supervised methods require large amounts of data to be curated as labeled data (i.e., with ground truth). Unsupervised methods do not require labeled data. Therefore, unsupervised learning can be applied to a wider range of datasets.

[5]The work presented in this chapter has been published in the following paper:
M. Ashrafuzzaman, S. Das, A. Jillepalli, Y. Chakhchoukh, and F. T. Sheldon. “Elliptic envelope for detecting stealthy false data injection attacks in the smart grid control systems,” in IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, December 2020.

While it may not be very difficult to collect data during normal operation of transmission systems, “attack” data is very rarely available because attacks do not happen often. In addition to that the process of curating the data to create labeled datasets is an onerous task. Instead of being trained on data labeled with ground truth, unsupervised models work by finding the hidden similarities in different data components and attempt to group similar data instances in regions or clusters. For bi-modal data like SE measurement data with sparse attacks, unsupervised models create one cluster fitting most of the non-attack data (major data) and treat instances lying far outside the cluster as anomalies. Therefore, unsupervised models are particularly well-suited to discover zero-day attacks never before encountered. Elliptic envelope algorithm is an unsupervised machine learning method that uses covariance estimation on Gaussian distribution data [96]. Elliptic envelope tries to make an elliptical cluster and fits the major class instances in that. Instances far away from the cluster are then considered as anomalies. Therefore, elliptic envelope is suitable for Gaussian SE measurement data with sparse SFDIA.

9.3 ELLIPTIC ENVELOPE METHOD

In this scheme, unsupervised machine learning method elliptic envelope is used as an anomaly detector to diagnose SFDIA on state estimation in power transmission systems. The elliptic envelope method models data as a high dimensional Gaussian distribution with possible covariances between data-features. It attempts to delineate an ellipse so that majority of the data instances fit into the ellipse. Data instances lying far outside the ellipse are then considered anomalies or outliers and, for the current context, are marked as an attack. The elliptic envelope method uses the FAST-minimum covariance determinant (FAST-MCD) [97] to estimate the shape and size of the ellipse. The FAST-MCD algorithm selects non-intersecting sub-sets of data and computes the mean μ, and the covariance matrix C, in each data-feature for each sub-set. The Mahalanobis distance, d_{MH}, a measure of the distance between a point P and a distribution D, is computed for each multidimensional data vector x, in each sub-set and the data are ordered in ascending order by d_{MH}. The Mahalanobis distance obtained from this estimate is used to define the threshold for determining outliers or anomalies. The Mahalanobis distance is defined by Mahalanobis [75] as:

$$d_{MH} = \sqrt{(x-\mu)^T C^{-1} (x-\mu)} \tag{9.1}$$

where C is the covariance matrix. If the covariance matrix is the identity matrix, then d_{MH} reduces to the Euclidean distance and to the normalized Euclidean distance if the covariance matrix is diagonal. In essence, the Mahalanobis distance measures how many σ (standard deviation) a data point is from the mean of a distribution. The FAST-MCD algorithm selects sub-sets from the original dataset, with small values of d_{MH}. Then computes mean, covariance, and the values of d_{MH} of the sub-sets. This procedure is iterated until the determinate of the covariance matrix converges. The covariance matrix with the smallest determinate from all sub-set forms an ellipse which encloses majority of the data.

An implementation of elliptic envelope method provided by the scikit-learn Python package [90] was used in this exercise.

9.4 MODEL SELECTION

To find an optimized elliptic envelope model, the models were trained with varying values for the hyper-parameter "contamination rate". Contamination rate represents the proportion of anomalies or outliers in the dataset. The contamination rate describes approximately how much of the data instances should sit outside of the enclosing high-dimensional ellipse that contains the majority of the data instances. In this model selection experiment, the values of contamination rate started at 0.001 and gradually increased to 0.5.

The dataset was split in 7:3 ratio into training subset and test subset retaining the same distribution of normal and attack data. To obtain robust models without over-fitting, 10-fold cross-validation over randomly divided sub-sets of training data during training of the models was used. Then, the test data was used for prediction and for measuring model performance.

TABLE 9.1: Evaluation metrics values for the best-performing elliptic envelope models for the six datasets.

Datasets	**Contamination Rate**	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**Elapsed Time**
14-Bus-AM1	0.03	0.7183	0.9432	0.7286	0.7279	0.9787	46.94s
14-Bus-AM2	0.035	0.6945	0.9433	0.6792	0.7105	0.9665	39.77s
14-Bus-AM3	0.035	0.6882	0.9391	0.6762	0.7007	0.9644	42.63s
57-Bus-AM1	0.035	0.6920	0.9418	0.6606	0.7265	0.9631	56.94s
57-Bus-AM2	0.035	0.7046	0.9432	0.6831	0.7275	0.9653	52.92s
57-Bus-AM3	0.035	0.6903	0.9408	0.6663	0.7161	0.9636	56.16s

9.5 RESULTS

Table 8.1 lists the best EE models for different datasets and their corresponding evaluation metrics values. Detail results of elliptic envelope model selection using different datasets and associated evaluation metrics values are given in Appendix A.2.

9.6 DISCUSSION OF RESULTS

In this section, the results from the evaluations with different elliptic envelope models are discussed. The observations from Tables A.19-A.24 and Table 9.1 are:

- The contamination rate value corresponding to the best-performing model is 0.03 for 14-Bus-AM1 dataset, and 0.035 for five other datasets.
- The best sensitivity value varies from 70.07% to 72.79% for the six datasets.
- The best specificity value varies from 96.44% to 97.87% for the six datasets.
- The models take on an average 45s to train.

9.7 SUMMARY

In this chapter, use of the security analytics framework with elliptic envelope in the model selection phase was described for detection of SFDI attacks. For the model selection process a wide range of hyper-parameter values were used for the elliptic envelope models. It is found that the best-performing elliptic envelope model can detect 73% attacks with a false alarm rate of 3%.

CHAPTER 10

CONCLUSION

10.1 PRIMARY RESULTS

The primary results of this dissertation include:

1. Successful use of security analytics to detect stealthy false data injection attacks on the static state estimation process used by power transmission control systems, demonstrated by using simulated datasets.

2. Achieving a 98.24% detection rate and a 1.25% false alarm rate using supervised artificial neural networks with 4 hidden layers and 1200 hidden units per layer. These results are among the best results in the related works in the literature.

3. Among the unsupervised models, achieving a 73% detection rate and a 3% false alarm rate using elliptic envelope model.

4. Detection rates are the same for datasets with different attack models and different bus systems.

10.2 DISSERTATION SUMMARY

Research has shown that intruders in the power transmission systems can surreptitiously change the RTU sensor measurements in such a way that can upend the situational awareness of the operators of power control centers and force them to act in ways that may cause electricity theft, changes in the electricity prices, or outages. These subtle changes in the state estimation measure data, called stealthy false data injection attacks, cannot be detected by the conventional measures in the power transmission systems. Very active research is underway to devise a mechanism to accurately, reliably, and timely detect these attacks.

This research aimed at developing a scheme to accurately and reliably detect SFDI attacks on the static state estimation in AC power transmission systems. The scheme developed and presented in this dissertation, is a security analytics framework that employs machine learning models to detect SFDI attacks on the static state estimation in AC power transmission systems. As part of the framework, a threat model was developed and three attack models were identified, based on which synthetic data simulating standard IEEE 14-bus and 57-bus systems using MATPOWER were generated. A software for managing machine learning model selection and evaluation was developed using the Python programming language and scikit-learn machine learning library. This software supports a number of supervised, unsupervised, and stacking ensemble (Section 7.4.2) methods. These models were parameter-tuned, trained and tested with the generated datasets, and evaluated.

The experiments to detect SFDIA were started with three supervised models, GLM, GBM, and DRF, and it was found that the results, with a detection rate of 72%, were not very encouraging. Then, stacking ensembles of supervised and unsupervised models were used with the hope that it will improve the detection rates. However, even the best results from the ensembles were also not very good, with the detection rate being 73%. The next attempt was to use artificial neural networks with wide ranges of hidden layers and hidden units. It was found that when trained with the right hyper-parameter settings (i.e., 4 hidden layers and 1200 hidden units per layer), ANN can detect 98.24% of the attacks with only 1.25% false alarm rate. This result is among the best results found in the literature for SFDI attack detection for simulated data using IEEE 14- and 57-Bus systems.

Since, unsupervised models do not need labeled data to train, it was also imperative to find out which unsupervised model performs the best with these datasets. It was found that the elliptic envelope model can detect 73% of the attacks with a false alarm rate of 3%. It was also found that the detection rates for all the methods used in this dissertation are the same for the datasets corresponding to different bus sizes and different attack models. This means that irrespective of whether the attacker compromised one sensor, multiple sensors, or all sensors, the machine learning methods will be able to equally detect the attacks.

Therefore, this dissertation shows that machine learning based security analytics can accurately and reliably detect SFDI attacks on the static state estimation in power transmission systems.

In addition to the security analytics framework, the software, and the results obtained with that, a survey of different types of false data injection attacks on different parts of the entire power grid and a taxonomy of such attacks were presented in the dissertation. Also presented was a survey of literature of the machine learning based approaches to detect SFDI attacks on the static state estimation of the power transmission systems. The ideas listed as future work below show that the research will open up a number of immediate research questions in this urgent and active area. The software and the datasets will be made available via GitHub so that the experiments can be reproduced and future works can be performed.

10.3 FUTURE WORK

The future work items of this research can be categorized as in the sub-sections below.

10.3.1 *Extending the Security Analytics Software*

- The security analytics software implemented does not have a graphical user interface. A user-friendly front-end will add to the completion and usability of the software.
- At this time the software supports GLM, GBM, DRF, DT, LR, NB, ANN, SVM, OCSVM with different kernels, ISOF, LOF, and EE methods. More methods can be added to the program easily.
- In addition to incorporating new methods, the software can be made configurable. An extensible and configurable framework will ensure that such a technique could be standardized for systematic use in transmission systems and other problem areas.
- The methods supported are all for time-discrete data analytics. Addition of time-series analytics methods like recurrent neural networks will enable the software to perform time-series analysis.
- Support for adversarial machine learning can be another useful future work. In adversarial machine learning, the input data is deliberately contaminated to deceive the ML model in predicting incorrectly. Adversarial machine learning has been shown to be useful for threat modeling, attack-defense scenario modeling, attack simulation, etc.

- Another potential future work is extending the ML software to support explanation-based learning (EBL).
- At this time, the software uses only RFC for feature selection. More feature reduction methods can be added.
- More visualization support, e.g., interactive graphs, 3-D graphs, etc., can be added.

10.3.2 *Further Research*

- It was seen in chapters 6 and 7 that the performance of all supervised methods for all the six datasets are essentially the same. This is not expected as different machine learning methods usually give somewhat different classification performance with the same dataset. An important future work will be to investigate why all the supervised methods resulted in the same performance metrics values. One way to undertake this investigation can be by using interpretable machine learning (IML) methods. Traditional machine learning methods use a black-box approach with the internal operations of the methods hidden to the human. IML explains a machine learning model in an understandable way for the human. Interpretability is the degree of measurement on how much a human can understand the reasons behind the ML models' decisions (i.e., predictions). Alternately, interpretability is the ability of understanding the decision-making policies of the machine-learned response function in order to explain the relationships between independent (input) and dependent (target or response) variables, ideally in a humanly interpretable manner. The interpretable models are created using white-box techniques. More information about interpretable machine learning can be found in [82]. In another approach to investigate the above issue of supervised models giving same results, datasets with new parameters can be generated and/or the datasets can be scrambled before employing the machine learning methods. This may uncover additional insight into the properties of datasets pertaining to stealthy false data injection attacks. Combining IML approach with this may result in formulating a science of SFDIA modeling and testing to ensure 360 degree fidelity and trustworthiness of the machine learning model predictions.

- This dissertation used one boosting (GBM) and one bagging (DRF) methods. None of these methods yielded good classification results. Boosting and Bagging are often thought of as meta-heuristics. Rather than treat them as separate algorithms, they might be explored as an additional option on the tests.

- In this research, the machine learning algorithms were run on limited-power desktop machines. It is anticipated that use of high-performance, parallel computing systems with graphical processing units (GPU) will reduce the training time drastically. Empirical data is needed to prove this assumption.

- This research used 14-bus and 57-bus system to validate the security analytics framework. Further testing with systems having large number of buses will shed light on the scalability of the framework. High-performance computing (HPC) system with GPUs will be needed for this investigation.

- This research shows 98% detection rate can be achieved for the synthetic datasets. Further research can be undertaken to investigate the ability of ML to detect subtler and smaller changes in the measurement data.

- There is a potentiality that unsupervised anomaly detection can detect never-before-seen attacks since unsupervised models do not need labeled data. This needs to be investigated further.

- This research investigated only time-discrete data. Further research is needed to take into account the correlation in power transmission data with seasons and time of the day. Correlated time-series data needs to be analyzed using ML methods like RNN and LSTM.

- This research dealt with measurement data associated with static state estimation. More and more power systems are now using PMUs and hence dynamic state estimation. It would be interesting to know how SFDI attacks impact dynamic SE and if machine learning can be used to detect SFDI attacks on dynamic state estimation.

- The ML software implemented is application-ignorant, i.e., the software can be used for analytics in many other areas, including security analytics of other power grid security issues, ICS and CPS security issues, computational biology, network security, medical analytics, etc.

- SFDI attacks are still limited to academic research. Except for the 2015 Ukraine attack, there is no other known instance of SFDI attack. Consequently, data corresponding to actual attacks are not available. However, it may be possible to use power system testbeds and real-time digital simulation (RTDS) to generate more realistic data, and use security analytics to investigate detection performance.

- Deployment of the trained models to find out how efficiently the models detect SFDI attacks, possibly using a power system testbed, is another future work.

- Only a conceptual model of a damage mitigation mechanism is presented in this dissertation (in Appendix B). This conceptual model can be fleshed out, implemented, and evaluated as another future work.

10.4 FINAL COMMENT

Stealthy false data injection attacks currently exist only in academic research labs. However, it is certainly possible that such an attack could be successful in real-life in causing power theft or even power system failure which could be deadly and extremely costly. Studies such as this one, help the community to understand more completely how detection, prevention, and countermeasure mechanisms of this type can play a role. Ideally, defense-in-depth or layered defenses should prevent the adversaries from spoofing the system to enable such SFDI attacks. However, there is no guarantee that adversaries will not be successful in penetrating layered defenses. Moreover, the SFDI attack is also a threat from the inside. Users, with proper credentials constitute a significant portion of the threat in this SFDI attack surface. Therefore, active research is necessary to devise even more effective mechanisms to quickly detect and deter such attacks. The research presented in this dissertation is a step towards achieving a higher confidence, more resilient energy delivery systems.

Bibliography

[1] CIA: cyberattack caused multiple-city blackout, January 2008. www.cnet.com/news/cia-cyberattack-caused-multiple-city-blackout/.

[2] A. Abur and A. Gomez-Exposito. *Power System State Estimation: Theory and Implementation*. CRC Press, New York, 2004. doi:10.1201/9780203913673.

[3] Saeed Ahmed, YoungDoo Lee, Seung-Ho Hyun, and Insoo Koo. Covert cyber assault detection in smart grid networks utilizing feature selection and Euclidean distance-based machine learning. *Applied Sciences*, 8(5):772–792, 2018. doi:10.3390/app8050772.

[4] Saeed Ahmed, Youngdoo Lee, Seung-Ho Hyun, and Insoo Koo. Feature selection–based detection of covert cyber deception assaults in smart grid communications networks using machine learning. *IEEE Access*, 6:27518–27529, 2018. doi:10.1109/ACCESS.2018.2835527.

[5] Saeed Ahmed, YoungDoo Lee, Seung-Ho Hyun, and Insoo Koo. Unsupervised machine learning-based detection of covert data integrity assault in smart grid networks utilizing isolation forest. *IEEE Transactions on Information Forensics and Security*, 14(10):2765–2777, 2019. doi:10.1109/TIFS.2019.2902822.

[6] Cristina Alcaraz and Javier Lopez. Wide-area situational awareness for critical infrastructure protection. *Computer*, 46(4):30–37, 2013. doi:10.1109/MC.2013.72.

[7] Oyeniyi Akeem Alimi, Khmaies Ouahada, and Adnan M Abu-Mahfouz. Real time security assessment of the power system using a hybrid support vector machine and multilayer perceptron neural network algorithms. *Sustainability*, 11(13):3586, 2019. doi:10.3390/su11133586.

[8] Adnan Anwar and Abdun Naser Mahmood. Stealthy and blind false injection attacks on scada ems in the presence of gross errors. In *2016 IEEE Power and Energy Society General Meeting (PESGM)*, pages 1–5. IEEE, 2016.

[9] Adnan Anwar, Abdun Naser Mahmood, and Mark Pickering. Modeling and performance evaluation of stealthy false data injection attacks on smart grid in the presence of corrupted measurements. *Journal of Computer and System Sciences*, 83(1):58–72, 2017.

[10] Adnan Anwar, Abdun Naser Mahmood, and Zahir Tari. Identification of vulnerable node clusters against false data injection attack in an AMI based smart grid. *Information Systems*, 53:201–212, 2015.

[11] Mohammad Ashrafuzzaman, Yacine Chakhchoukh, Ananth Jillepalli, Predrag Tosic, Daniel Conte de Leon, Frederick T Sheldon, and Brian Johnson. Detecting stealthy false data injection attacks in power grids using deep learning. In *Wireless Communications and Mobile Computing Conference (IWCMC), 14th International*, pages 219–225. IEEE, 2018. doi:10.1109/IWCMC.2018.8450487.

[12] Abdelrahman Ayad, Hany EZ Farag, Amr Youssef, and Ehab F El-Saadany. Detection of false data injection attacks in smart grids using recurrent neural networks. In *2018 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5. IEEE, 2018. `doi:10.1109/ISGT.2018.8403355`.

[13] Gustavo E A P A Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, 6(1):20–29, 2004. `doi:10.1145/1007730.1007735`.

[14] Omar Ali Beg, Taylor T Johnson, and Ali Davoudi. Detection of false-data injection attacks in cyber-physical DC microgrids. *IEEE Transactions on industrial Informatics*, 13(5):2693–2703, 2017.

[15] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 93–104, 2000. `doi:10.1145/342009.335388`.

[16] Eric Byres. The air gap: SCADA's enduring security myth. *Communications of the ACM*, 56(8):29–31, 2013. `doi:10.1145/2492007.2492018`.

[17] Mario R Camana-Acosta, Saeed Ahmed, Carla E Garcia, and Insoo Koo. Extremely randomized trees-based scheme for stealthy cyber-attack detection in smart grid networks. *IEEE Access*, 8:19921–19933, 2020. `doi:10.1109/ACCESS.2020.2968934`.

[18] Yacine Chakhchoukh and Hideaki Ishii. Coordinated cyber-attacks on the measurement function in hybrid state estimation. *IEEE Transactions on Power Systems*, 30(5):2487–2497, September 2015. `doi:10.1109/TPWRS.2014.2357182`.

[19] Yacine Chakhchoukh, Song Liu, Masashi Sugiyama, and Hideaki Ishii. Statistical outlier detection for diagnosis of cyber attacks in power state estimation. In *2016 IEEE Power and Energy Society General Meeting (PESGM)*, pages 1–5. IEEE, 2016.

[20] Gu Chaojun, Panida Jirutitijaroen, and Mehul Motani. Detecting false data injection attacks in AC state estimation. *IEEE Transactions on Smart Grid*, 6(5):2476–2483, 2015. `doi:10.1109/TSG.2015.2388545`.

[21] Jiongcong Chen, Gaoqi Liang, Zexiang Cai, Chunchao Hu, Yan Xu, Fengji Luo, and Junhua Zhao. Impact analysis of false data injection attacks on power system static security assessment. *Journal of Modern Power Systems and Clean Energy*, 4(3):496–505, 2016.

[22] Dae-Hyun Choi and Le Xie. Ramp-induced data attacks on look-ahead dispatch in real-time power markets. *IEEE Transactions on Smart Grid*, 4(3):1235–1243, 2013.

[23] Nello Christianini and John Shawe-Taylor. Support vector machines and other kernel-based learning methods. *Cambridge UP*, 2000.

[24] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. doi:10.1007/BF00994018.

[25] George Cybenko and Carl E Landwehr. Security analytics and measurements. *IEEE Security & Privacy*, 10(3):5–8, 2012. doi:10.1109/MSP.2012.75.

[26] Saikat Das, Deepak Venugopal, and Sajjan Shiva. A holistic approach for detecting DDoS attacks by using ensemble unsupervised machine learning. In *Future of Information and Communication Conference*, pages 721–738. Springer, 2020. doi:10.1007/978-3-030-39442-4_53.

[27] Ruilong Deng, Peng Zhuang, and Hao Liang. False data injection attacks against state estimation in power distribution systems. *IEEE Transactions on Smart Grid*, 10(3):2871–2881, 2018.

[28] Thomas G Dietterich. Ensemble methods in machine learning. In *International Workshop on Multiple Classifier Systems*, pages 1–15. Springer, 2000. doi:10.1007/3-540-45014-9_1.

[29] Mohammad Esmalifalak, Lanchao Liu, Nam Nguyen, Rong Zheng, and Zhu Han. Detecting stealthy false data injection using machine learning in smart grid. *IEEE Systems Journal*, 11(3):1644–1652, 2017. doi:10.1109/JSYST.2014.2341597.

[30] Mohammad Esmalifalak, Huy Nguyen, Rong Zheng, and Zhu Han. Stealth false data injection using independent component analysis in smart grid. In *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 244–248. IEEE, 2011. doi:10.1109/SmartGridComm.2011.6102326.

[31] Mohammad Esmalifalak, Ge Shi, Zhu Han, and Lingyang Song. Bad data injection attack and defense in electricity market using game theory study. *IEEE Transactions on Smart Grid*, 4(1):160–169, 2013.

[32] S Armina Foroutan and Farzad R Salmasi. Detection of false data injection attacks against state estimation in smart grids based on a mixture Gaussian distribution learning method. *IET Cyber-Physical Systems: Theory & Applications*, 2017. doi:10.1049/iet-cps.2017.0013.

[33] Mehdi Ganjkhani, Seyedeh Narjes Fallah, Sobhan Badakhshan, Shahaboddin Shamshirband, and Kwok-wing Chau. A novel detection algorithm to identify false data injection attacks on power system state estimation. *Energies*, 12(11):2209, 2019.

[34] Annarita Giani, Eilyan Bitar, Manuel Garcia, Miles McQueen, Pramod Khargonekar, and Kameshwar Poolla. Smart grid data integrity attacks. *IEEE Transactions on Smart Grid*, 4(3):1244–1253, 2013.

[35] Jairo Giraldo, Esha Sarkar, Alvaro Cardenas, Michail Maniatakos, and Murat Kantarcioglu. Security and privacy in cyber-physical systems: A survey of surveys. *IEEE Design & Test, (2017)*, 2017. doi:10.1109/MDAT.2017.2709310.

[36] David Grochocki, Jun Ho Huh, Robin Berthier, Rakesh Bobba, William H Sanders, Alvaro A Cárdenas, and Jorjeta G Jetcheva. AMI threats, intrusion detection requirements and deployment recommendations. In *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*, pages 395–400. IEEE, 2012.

[37] Zhitao Guan, Nan Sun, Yue Xu, and Tingting Yang. A comprehensive survey of false data injection in smart grid. *International Journal of Wireless and Mobile Computing*, 8(1):27–33, 2015.

[38] Yonghe Guo, Chee-Wooi Ten, and Panida Jirutitijaroen. Online data validation for distribution operations against cybertampering. *IEEE Transactions on Power Systems*, 29(2):550–560, 2013.

[39] Mohamed Hamlich, Abdelkarim El Khantach, and Tarik Ibn Ziad. Machine learning methods against false data injection in smart grid. *International Journal of Reasoning-based Intelligent Systems*, 2(1), 2020. `doi:10.1504/IJRIS.2020.104991`.

[40] Jianye Hao, Eunsuk Kang, Jun Sun, Zan Wang, Zhaopeng Meng, Xiaohong Li, and Zhong Ming. An adaptive markov strategy for defending smart grid false data injection from malicious attackers. *IEEE Transactions on Smart Grid*, 9(4):2398–2408, 2016.

[41] Jinping Hao, Robert J Piechocki, Dritan Kaleshi, Woon Hau Chin, and Zhong Fan. Sparse malicious false data injection attacks and defense mechanisms in smart grids. *IEEE Transactions on Industrial Informatics*, 11(5):1–12, 2015. `doi:10.1109/TII.2015.2475695`.

[42] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag, 2nd edition, 2008.

[43] Youbiao He, Gihan J Mendis, and Jin Wei. Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism. *IEEE Transactions on Smart Grid*, 2017. `doi:10.1109/TSG.2017.2703842`.

[44] R. Hecht-Nielsen. Theory of the backpropagation neural network. In *International Joint IEEE Conference on Neural Networks*, pages 593–605, 1989. `doi:10.1109/IJCNN.1989.118638`.

[45] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[46] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[47] Mia Hubert and Michiel Debruyne. Minimum covariance determinant. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):36–43, 2010. `doi:10.1002/wics.61`.

[48] Gabriela Hug and Joseph Andrew Giampapa. Vulnerability assessment of AC state estimation with respect to false data injection cyber-attacks. *IEEE Transactions on Smart Grid*, 3(3):1362–1370, 2012. doi:10.1109/TSG.2012.2195338.

[49] Liyan Jia, Robert J Thomas, and Lang Tong. Impacts of malicious data on real-time price of electricity market operations. In *2012 45th Hawaii International Conference on System Sciences*, pages 1907–1914. IEEE, 2012.

[50] Xichen Jiang, Jiangmeng Zhang, Brian J Harding, Jonathan J Makela, and Alejandro D Domı´nguez-Garcı´a. Spoofing GPS receiver clock offset of phasor measurement units. *IEEE Transactions on Power Systems*, 28(3):3253–3262, 2013.

[51] Ananth A Jillepalli, Daniel Conte de Leon, Brian K Johnson, Yacine Chakhchoukh, Ibukun A Oyewumi, Mohammad Ashrafuzzaman, Frederick T Sheldon, Jim Alves-Foss, and Michael A Haney. METICS: A holistic cyber physical system model for ieee 14-bus power system security. In *2018 13th International Conference on Malicious and Unwanted Software (MALWARE)*, pages 95–102. IEEE, 2018.

[52] Vassilis Kekatos, Georgios B Giannakis, and Ross Baldick. Grid topology identification using electricity prices. In *2014 IEEE PES General Meeting| Conference & Exposition*, pages 1–5. IEEE, 2014. doi:10.1109/PESGM.2014.6939474.

[53] Shehroz S Khan and Michael G Madden. A survey of recent trends in one class classification. In *Irish Conference on Artificial Intelligence and Cognitive Science*, pages 188–197. Springer, 2009. doi:10.1007/978-3-642-17080-5_21.

[54] Jinsub Kim and Lang Tong. On topology attack of a smart grid: Undetectable attacks and countermeasures. *IEEE Journal on Selected Areas in Communications*, 31(7):1294–1305, 2013.

[55] Jinsub Kim, Lang Tong, and Robert J Thomas. Data framing attack on state estimation. *IEEE Journal on selected areas in communications*, 32(7):1460–1470, 2014. doi:10.1109/JSAC.2014.2332032.

[56] Oliver Kosut, Liyan Jia, Robert J Thomas, and Lang Tong. Malicious data attacks on the smart grid. *IEEE Transactions on Smart Grid*, 2(4):645–658, 2011. doi:10.1109/TSG.2011.2163807.

[57] Mehmet Necip Kurt, Oyetunji Ogundijo, Chong Li, and Xiaodong Wang. Online cyber-attack detection in smart grid: A reinforcement learning approach. *IEEE Transactions on Smart Grid*, 10(5):5174–5185, 2018. doi:10.1109/TSG.2018.2878570.

[58] Kingsly Leung and Christopher Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science*, volume 38, pages 333–342, 2005.

[59] Yan Li, Peng Zhang, Liang Zhang, and Bing Wang. Active synchronous detection of deception attacks in microgrid control systems. *IEEE Transactions on Smart Grid*, 8(1):373–375, 2017.

[60] Zhiyi Li, Mohammad Shahidehpour, Ahmed Alabdulwahab, and Abdullah Abusorrah. Bilevel model for analyzing coordinated cyber-physical attacks on power systems. *IEEE Transactions on Smart Grid*, 7(5):2260–2272, 2015.

[61] Gaoqi Liang, Steven R Weller, Junhua Zhao, Fengji Luo, and Zhao Yang Dong. The 2015 Ukraine blackout: Implications for false data injection attacks. *IEEE Transactions on Power Systems*, 32(4):3317–3318, 2017. `doi:10.1109/TPWRS.2016.2631891`.

[62] Gaoqi Liang, Junhua Zhao, Fengji Luo, Steven R Weller, and Zhao Yang Dong. A review of false data injection attacks against modern power systems. *IEEE Transactions on Smart Grid*, 8(4):1630–1638, 2016. `doi:10.1109/TSG.2015.2495133`.

[63] IH Lim, S Hong, MS Choi, SJ Lee, TW Kim, SW Lee, and BN Ha. Security protocols against cyber attacks in the distribution automation system. *IEEE Transactions on Power Delivery*, 25(1):448–455, 2010.

[64] Jie Lin, Wei Yu, Xinyu Yang, Guobin Xu, and Wei Zhao. On false data injection attacks against distributed energy routing in smart grid. In *Proceedings of the 2012 IEEE/ACM Third International Conference on Cyber-Physical Systems*, pages 183–192. IEEE Computer Society, 2012.

[65] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):1–39, 2012. `doi:10.1145/2133360.2133363`.

[66] Mengxiang Liu, Chengcheng Zhao, Ruilong Deng, Peng Cheng, Wenhai Wang, and Jiming Chen. Nonzero-dynamics stealthy attack and its impacts analysis in DC microgrids. In *2019 American Control Conference (ACC)*, pages 3922–3927. IEEE, 2019.

[67] Xuan Liu, Zhen Bao, Dan Lu, and Zuyi Li. Modeling of local false data injection attacks with reduced network information. *IEEE Transactions on Smart Grid*, 6(4):1686–1696, 2015. `doi:10.1109/TSG.2015.2394358`.

[68] Xuan Liu, Zhiyi Li, Xingdong Liu, and Zuyi Li. Masking transmission line outages via false data injection attacks. *IEEE Transactions on Information Forensics and Security*, 11(7):1592–1602, 2016.

[69] Xuan Liu and Zuyi Li. Local load redistribution attacks in power systems with incomplete network information. *IEEE Transactions on Smart Grid*, 5(4):1665–1676, 2014. `doi:10.1109/TSG.2013.2291661`.

[70] Xuan Liu and Zuyi Li. False data attacks against ac state estimation with incomplete network information. *IEEE Transactions on smart grid*, 8(5):2239–2248, 2016.

[71] Xuan Liu and Zuyi Li. Local topology attacks in smart grids. *IEEE Transactions on Smart Grid*, 8(6):2617–2626, 2016.

[72] Xuan Liu and Zuyi Li. False data attack models, impact analyses and defense strategies in the electricity grid. *The Electricity Journal*, 30:35–42, 2017. doi:10.1016/j.tej.2017.04.001.

[73] Y. Liu, M. K. Reiter, and P. Ning. False data injection attacks against state estimation in electric power grids. In *Proc. 16th ACM Conf. on Computer and Communications Security*, pages 21–32, 2009.

[74] Yao Liu, Peng Ning, and Michael K Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security*, 14(1):13:1–13:33, 2011. doi:10.1145/1952982.1952995.

[75] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. In *Proceedings of the National Institute of Sciences of India*, volume 2:1, pages 49–55. National Institute of Science of India, 1936.

[76] Stephen Marsland. *Machine Learning: An Algorithmic Perspective*. CRC press, 2015.

[77] Stephen McLaughlin, Charalambos Konstantinou, Xueyang Wang, Lucas Davi, Ahmad-Reza Sadeghi, Michail Maniatakos, and Ramesh Karri. The cybersecurity landscape in industrial control systems. *Proceedings of the IEEE*, 104(5):1039–1057, 2016. doi:10.1109/JPROC.2015.2512235.

[78] Stephen McLaughlin, Dmitry Podkuiko, and Patrick McDaniel. Energy theft in the advanced metering infrastructure. In *International Workshop on Critical Information Infrastructures Security*, pages 176–187. Springer, 2009.

[79] Mostafa Mohammadpourfard, Ashkan Sami, and Ali Reza Seifi. A statistical unsupervised method against false data injection attacks: A visualization-based approach. *Expert Systems with Applications*, 84:242–261, 2017.

[80] Mostafa Mohammadpourfard, Yang Weng, Mykola Pechenizkiy, Mohsen Tajdinian, and Behnam Mohammadi-Ivatloo. Ensuring cybersecurity of smart grid against data integrity attacks under concept drift. *International Journal of Electrical Power & Energy Systems*, 119:105947, 2020. doi:10.1016/j.ijepes.2020.105947.

[81] Amir-Hamed Mohsenian-Rad and Alberto Leon-Garcia. Distributed internet-based load altering attacks against smart power grids. *IEEE Transactions on Smart Grid*, 2(4):667–674, 2011. doi:10.1109/TSG.2011.2160297.

[82] Christoph Molnar. *Interpretable Machine Learning*. Lean Publishing, 2020.

[83] Nour Moustafa, Benjamin Turnbull, and Kim-Kwang Raymond Choo. An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. *IEEE Internet of Things Journal*, 6(3):4815–4830, 2018. doi:10.1109/JIOT.2018.2871719.

[84] Christian Moya, Chih-Che Sun, Jiankang Wang, and Chen-Ching Liu. Defensing against measurement attacks on sub-transmission level. In *Power and Energy Society General Meeting (PESGM), 2016*, pages 1–5. IEEE, 2016. doi:10.1002/9781118755471.sgd055.

[85] Ahmed S Musleh, Guo Chen, and Zhao Yang Dong. A survey on the detection algorithms for false data injection attacks in smart grids. *IEEE Transactions on Smart Grid*, 11(3):2218–2234, 2019.

[86] J. A. Nelder and R. W. M. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384, 1972.

[87] Xiangyu Niu, Jiangnan Li, Jinyuan Sun, and Kevin Tomsovic. Dynamic detection of false data injection attack in smart grid using deep learning. In *2019 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–6. IEEE, 2019. doi:10.1109/ISGT.2019.8791598.

[88] Mete Ozay, Inaki Esnaola, Fatos Tunay Yarman Vural, Sanjeev R Kulkarni, and H Vincent Poor. Machine learning methods for attack detection in the smart grid. *IEEE Transactions on Neural Networks and Learning Systems*, 27(8):1773–1786, 2015. doi:10.1109/TNNLS.2015.2404803.

[89] Sarita Paudel, Paul Smith, and Tanja Zseby. Attack models for advanced persistent threats in smart grid wide area monitoring. In *Proceedings of the 2nd Workshop on Cyber-Physical Security and Resilience in Smart Grids*, pages 61–66. ACM, 2017. doi:10.1145/3055386.3055390.

[90] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[91] Robi Polikar. Ensemble learning. In *Ensemble Machine Learning*, pages 1–34. Springer, 2012. doi:10.1007/978-1-4419-9326-7_1.

[92] Ponemon Institute. Cybersecurity in operational technology: 7 insights you need to know, April 2019. accessed on August 16, 2020. URL: https://lookbook.tenable.com/ponemonotreport/ponemon-OT-report.

[93] Md Ashfaqur Rahman and Hamed Mohsenian-Rad. False data injection attacks with incomplete information against smart power grids. In *2012 IEEE Global Communications Conference*, pages 3153–3158. IEEE, 2012. doi:10.1109/GLOCOM.2012.6503599.

[94] Md Ashfaqur Rahman and Hamed Mohsenian-Rad. False data injection attacks against nonlinear state estimation in smart power grids. In *2013 IEEE Power & Energy Society General Meeting*, pages 1–5. IEEE, 2013.

[95] Mohammad Ashiqur Rahman, Ehab Al-Shaer, and Rajesh Kavasseri. Impact analysis of topology poisoning attacks on economic operation of the smart power grid. In *2014 IEEE 34th International Conference on Distributed Computing Systems*, pages 649–659. IEEE, 2014.

[96] Peter J Rousseeuw. Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880, 1984.

[97] Peter J Rousseeuw and Katrien Van Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223, 1999.

[98] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015. doi:10.1016/j.neunet.2014.09.003.

[99] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. In *Advances in Neural Information Processing Systems*, pages 582–588, 2000.

[100] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, 2009. doi:10.1016/j.ipm.2009.03.002.

[101] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density Ratio Estimation in Machine Learning*. Cambridge University Press, 2012.

[102] Rui Tan, Hoang Hai Nguyen, Eddy Foo, Xinshu Dong, David KY Yau, Zbigniew Kalbarczyk, Ravishankar K Iyer, and Hoay Beng Gooi. Optimal false data injection attack against automatic generation control in power grids. In *Proceedings of the 7th International Conference on Cyber-Physical Systems*, page 2. IEEE Press, 2016. doi:10.1109/ICCPS.2016.7479109.

[103] Rui Tan, Hoang Hai Nguyen, Eddy YS Foo, David KY Yau, Zbigniew Kalbarczyk, Ravishankar K Iyer, and Hoay Beng Gooi. Modeling and mitigating impact of false data injection attacks on automatic generation control. *IEEE Transactions on Information Forensics and Security*, 12(7):1609–1624, 2017. doi:10.1109/TIFS.2017.2676721.

[104] Song Tan, Debraj De, Wen-Zhan Song, Junjie Yang, and Sajal K Das. Survey of security advances in smart grid: A data driven approach. *IEEE Communications Surveys & Tutorials*, 19(1):397–422, 2017. doi:10.1109/COMST.2016.2616442.

[105] Mini S Thomas and John Douglas McDonald. *Power System SCADA and Smart Grids*. CRC press, 2015.

[106] University of Washington. *Power System Test Case Archive*. [Online]. Available: http://www.ee.washington.edu/research/pstca/, 2018.

[107] Michel Verleysen and Damien François. The curse of dimensionality in data mining and time series prediction. In *International Work-Conference on Artificial Neural Networks*, pages 758–770. Springer, 2005. doi:10.1007/11494669_93.

[108] Defu Wang, Xiaojuan Wang, Yong Zhang, and Lei Jin. Detection of power grid disturbances and cyber-attacks based on machine learning. *Journal of Information Security and Applications*, 46:42–52, 2019. doi:10.1016/j.jisa.2019.02.008.

[109] Huaizhi Wang, Jiaqi Ruan, Guibin Wang, Bin Zhou, Yitao Liu, Xueqian Fu, and Jian-Chun Peng. Deep learning based interval state estimation of AC smart grids against sparse cyber attacks. *IEEE Transactions on Industrial Informatics*, 2018. doi:10.1109/TII.2018.2804669.

[110] Jingxuan Wang, Wenting Tu, Lucas CK Hui, Siu-Ming Yiu, and Eric Ke Wang. Detecting time synchronization attacks in cyber-physical systems with machine learning techniques. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 2246–2251. IEEE, 2017. doi:10.1109/ICDCS.2017.25.

[111] Shaocheng Wang and Wei Ren. Stealthy false data injection attacks against state estimation in power systems: Switching network topologies. In *Proceedings of the 2014 American Control Conference*. IEEE, 2014. doi:10.1109/ACC.2014.6858904.

[112] Yi Wang, Mahmoud Amin, Jian Fu, and Heba Moussa. A novel data analytical approach for false data injection cyber-physical attack mitigation in smart grids. *IEEE Access*, 2017. doi:10.1109/ACCESS.2017.2769099.

[113] John Warsinske. *The Official (ISC)^2 CISSP CBK Reference*. Sybex, 2019.

[114] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[115] Yingmeng Xiang, Zhilu Ding, Yichi Zhang, and Lingfeng Wang. Power system reliability evaluation considering load redistribution attacks. *IEEE Transactions on Smart Grid*, 8(2):889–901, 2016.

[116] Yingmeng Xiang, Lingfeng Wang, and Nian Liu. Coordinated attacks on electric power systems in a cyber-physical environment. *Electric Power Systems Research*, 149:156–168, 2017. doi:10.1016/j.epsr.2017.04.023.

[117] Le Xie, Yilin Mo, and Bruno Sinopoli. False data injection attacks in electricity markets. In *2010 First IEEE International Conference on Smart Grid Communications*, pages 226–231. IEEE, 2010.

[118] Le Xie, Yilin Mo, and Bruno Sinopoli. Integrity data attacks in power market operations. *IEEE Transactions on Smart Grid*, 2(4):659–666, 2011. doi:10.1109/TSG.2011.2161892.

[119] Ruzhi Xu, Dawei Chen, and Rui Wang. Data integrity attack detection for node voltage in cyber-physical power system. *Arabian Journal for Science and Engineering*, 2020. doi:10.1007/s13369-020-04813-y.

[120] J. Yan, B. Tang, and H. He. Detection of false data attacks in smart grid with supervised learning. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 1395–1402, July 2016. doi:10.1109/IJCNN.2016.7727361.

[121] Can Yang, Yong Wang, Yuhao Zhou, Jiongming Ruan, Wei Liu, et al. False data injection attacks detection in power system using machine learning method. *Journal of Computer and Communications*, 6(11):276, 2018. doi:10.4236/jcc.2018.611025.

[122] Liqun Yang, Yuancheng Li, and Zhoujun Li. Improved-ELM method for detecting false data attack in smart grid. *International Journal of Electrical Power & Energy Systems*, 91(Supplement C):183 – 191, 2017. doi:10.1016/j.ijepes.2017.03.011.

[123] Qingyu Yang, Jie Yang, Wei Yu, Dou An, Nan Zhang, and Wei Zhao. On false data-injection attacks against power system state estimation: Modeling and countermeasures. *IEEE Transactions on Parallel and Distributed Systems*, 25(3):717–729, 2014. doi:10.1109/TPDS.2013.92.

[124] Zong-Han Yu and Wen-Long Chin. Blind false data injection attack using PCA approximation method in smart grid. *IEEE Transactions on Smart Grid*, 6(3):1219–1226, 2015. doi:10.1109/TSG.2014.2382714.

[125] Yanling Yuan, Zuyi Li, and Kui Ren. Modeling load redistribution attacks in power systems. *IEEE Transactions on Smart Grid*, 2(2):382–390, 2011.

[126] Yanling Yuan, Zuyi Li, and Kui Ren. Quantitative analysis of load redistribution attacks in power systems. *IEEE Transactions on Parallel and Distributed Systems*, 23(9):1731–1738, 2012.

[127] Heng Zhang, Wenchao Meng, Junjian Qi, Xiaoyu Wang, and Wei Xing Zheng. Distributed load sharing under false data injection attack in an inverter-based microgrid. *IEEE Transactions on Industrial Electronics*, 66(2):1543–1551, 2018.

[128] Xialei Zhang, Xinyu Yang, Jie Lin, and Wei Yu. On false data injection attacks against the dynamic microgrid partition in the smart grid. In *2015 IEEE International Conference on Communications (ICC)*, pages 7222–7227. IEEE, 2015.

[129] Xiaocai Zhang, Zhixun Zhao, Yi Zheng, and Jinyan Li. Prediction of taxi destinations using a novel data embedding method and ensemble learning. *IEEE Transactions on Intelligent Transportation Systems*, 2019. doi:10.1109/TITS.2018.2888587.

[130] Yichi Zhang, Lingfeng Wang, Yingmeng Xiang, and Chee-Wooi Ten. Power system reliability evaluation with SCADA cybersecurity considerations. *IEEE Transactions on Smart Grid*, 6(4):1707–1721, July 2015. doi:10.1109/TSG.2015.2396994.

[131] Zhenghao Zhang, Shuping Gong, Aleksandar D Dimitrovski, and Husheng Li. Time synchronization attack in smart grid: Impact and analysis. *IEEE Transactions on Smart Grid*, 4(1):87–98, 2013.

[132] Ray Daniel Zimmerman, Carlos Edmundo Murillo-Sánchez, and Robert John Thomas. MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26(1):12–19, 2011. doi:10.1109/TPWRS.2010.2051168.

APPENDIX A

DETAILS OF THE RESULTS FROM CHAPTERS 7-9

A.1 RESULTS OF STACKING ENSEMBLE EXPERIMENTS

In this section, the performance metrics values for all six datasets from the experiments with the ensemble framework in Chapter 7 are presented in tables, graphs and ROC curves.

The evaluation metric values for supervised and unsupervised individual and ensemble models for 14-bus-AM1 dataset are tabulated in Tables A.1 and A.2. The corresponding bar-graphs are shown in Figures A.1 and A.2. The ROC curves are shown in Figures A.3 and A.4.

TABLE A.1: Evaluation metrics values for supervised individual and ensemble models using the 14-bus-AM1 dataset.

Models	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**ROC AUC**
LR	0.8439	0.8931	0.9991	0.7304	0.9997	0.8639
NB	0.8439	0.8931	0.9991	0.7304	0.9997	0.8081
NN	0.8439	0.8931	0.9991	0.7304	0.9997	0.8650
DT	0.8438	0.8930	0.9991	0.7302	0.9997	0.8797
SVM	0.8439	0.8931	0.9991	0.7304	0.9997	0.8642
Ens_MV	0.8439	0.8931	0.9991	0.7304	0.9997	–
Ens_LR	0.8472	0.8961	0.9993	0.7353	0.9997	0.8675
En_NB	0.8472	0.8961	0.9993	0.7353	0.9997	0.8675
Ens_NN	0.8472	0.8961	0.9993	0.7353	0.9997	0.8675
Ens_DT	0.8472	0.8961	0.9993	0.7353	0.9997	0.8675
Ens_SVM	0.8472	0.8961	0.9993	0.7353	0.9997	0.8675

TABLE A.2: Evaluation metrics values for unsupervised individual and ensemble models using the 14-bus-AM1 dataset.

Models	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**ROC AUC**
OCSVM_P	0.0834	0.4549	0.0494	0.2661	0.4743	0.3702
LOF	0.1388	0.8586	0.1604	0.1223	0.9343	0.5283
ISOF	0.3781	0.7939	0.2630	0.6722	0.8065	0.7393
EE	0.6318	0.9214	0.5606	0.7237	0.9418	0.8327
OCSVM_L	0.1731	0.5000	0.1023	0.5617	0.4938	0.5277
Ens_MV	0.4375	0.8892	0.4150	0.4626	0.9331	-
Ens_LR	0.6409	0.9394	0.6739	0.6111	0.9713	0.7912
Ens_NB	0.5502	0.9034	0.4679	0.6676	0.9264	0.7969
Ens_NN	0.6218	0.9216	0.5428	0.7278	0.9505	0.8341
Ens_DT	0.6615	0.9407	0.6689	0.6544	0.9686	0.8114
Ens_SVM	0.6615	0.9407	0.6689	0.6544	0.9686	0.8114

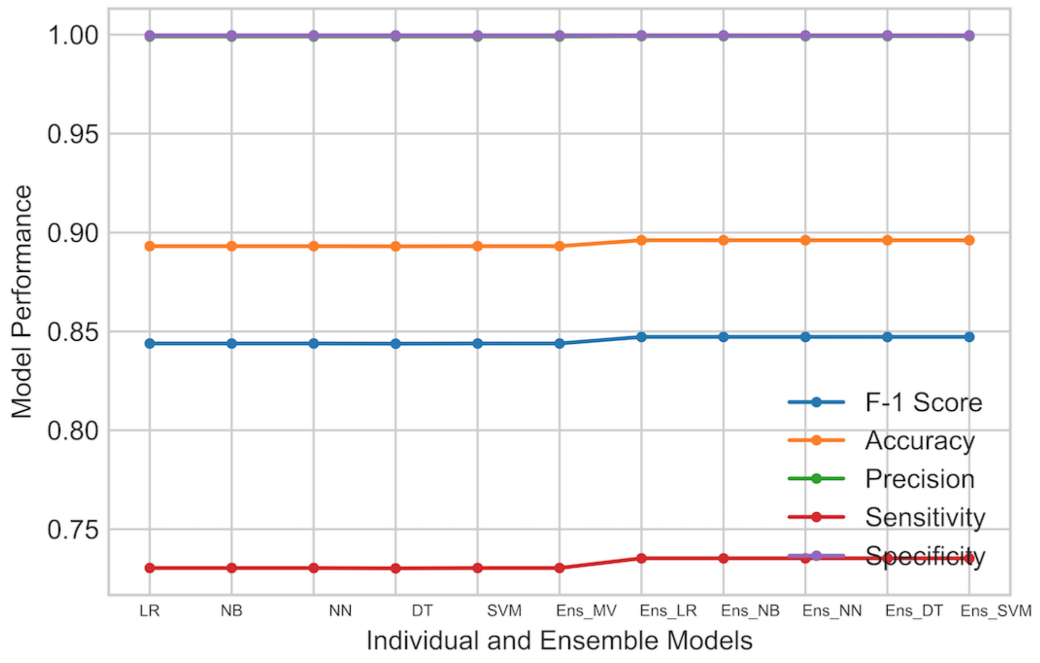


FIGURE A.1: Graph of the evaluation metrics values for supervised individual and ensemble models for the 14-bus-AM1 dataset.

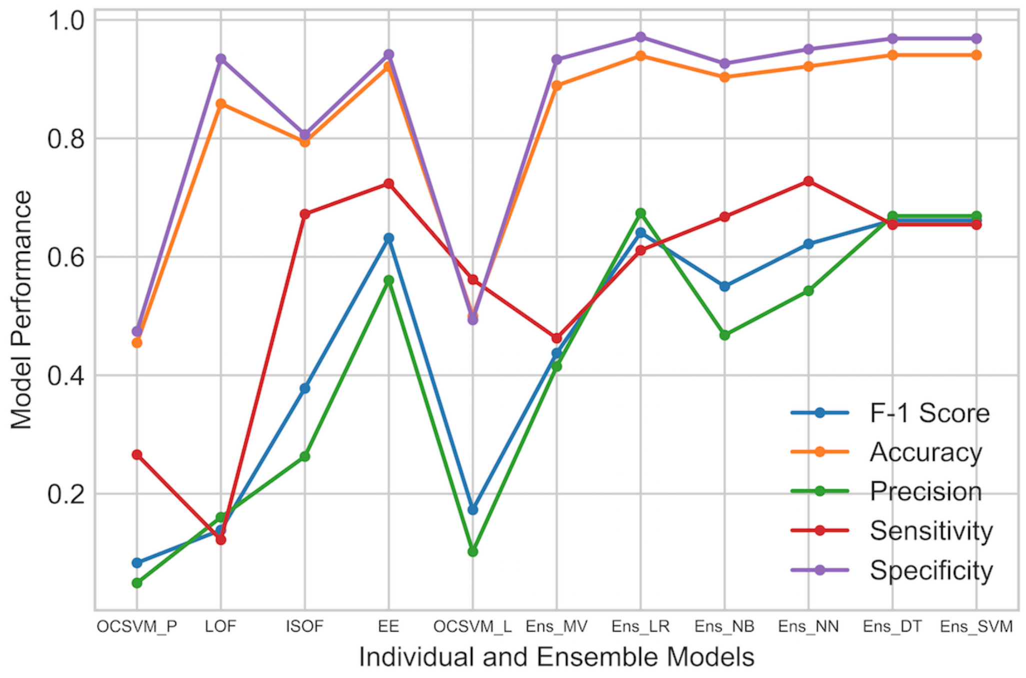


FIGURE A.2: Graph of the evaluation metrics values for unsupervised individual and ensemble models for the 14-bus-AM1 dataset.

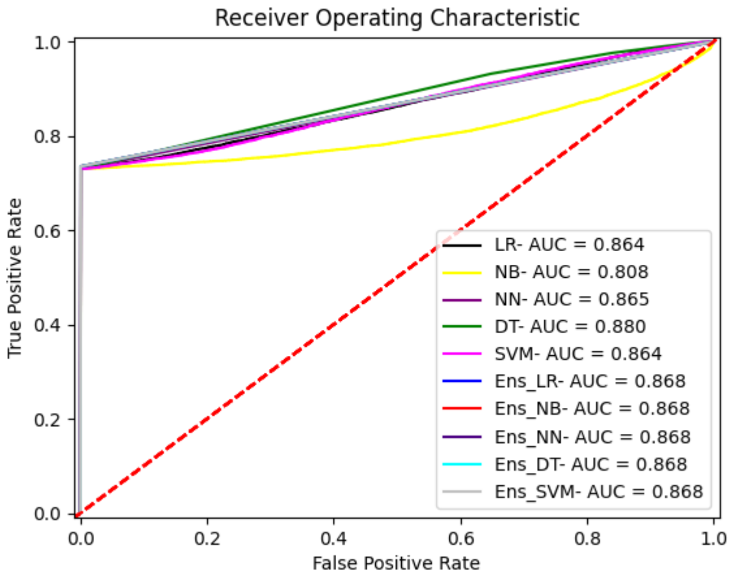


FIGURE A.3: ROC curves for the supervised individual and ensemble models for 14-bus-AM1 dataset.

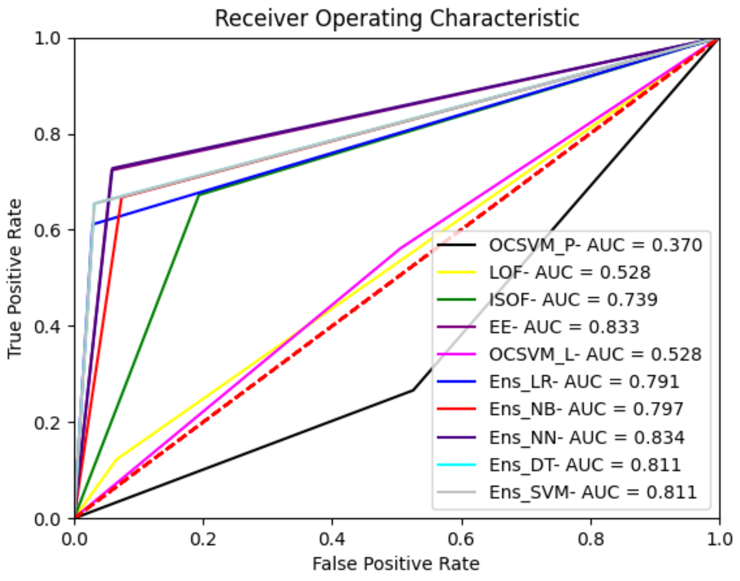


FIGURE A.4: ROC curves for the unsupervised individual and ensemble models for 14-bus-AM1 dataset.

The evaluation metric values for supervised and unsupervised individual and ensemble models for 14-bus-AM2 dataset in Tables A.3 and A.4. The corresponding bar-graphs are shown in Figures A.5 and A.6. The ROC curves are shown in Figures A.7 and A.8.

TABLE A.3: Evaluation metrics values for supervised individual and ensemble models using the 14-bus-AM2 dataset.

Models	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC
LR	0.8472	0.8913	0.9989	0.7356	0.9987	0.8661
NB	0.8472	0.8913	0.9989	0.7356	0.9987	0.8115
NN	0.8472	0.8913	0.9989	0.7354	0.9987	0.8951
DT	0.8472	0.8913	0.9989	0.7353	0.9987	0.8842
SVM	0.8472	0.8913	0.9989	0.7356	0.9987	0.8699
Ens_MV	0.8472	0.8913	0.9989	0.7356	0.9987	0.8684
Ens_LR	0.8477	0.8911	0.9991	0.7361	0.9989	0.8686
Ens_NB	0.8477	0.8911	0.9991	0.7361	0.9989	0.8683
Ens_NN	0.8477	0.8911	0.9991	0.7361	0.9989	0.8681
Ens_DT	0.8477	0.8911	0.9991	0.7361	0.9989	0.8682
Ens_SVM	0.8477	0.8911	0.9991	0.7361	0.9989	0.8684

TABLE A.4: Evaluation metrics values for unsupervised individual and ensemble models using the 14-bus-AM2 dataset.

Models	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC
OCSVM_P	0.0500	0.4611	0.0301	0.1501	0.4941	0.3221
LOF	0.0511	0.7105	0.0357	0.0832	0.7762	0.4291
ISOF	0.2582	0.5401	0.1519	0.8547	0.5073	0.6811
EE	0.6534	0.9379	0.7031	0.6107	0.9728	0.7923
OCSVM_L	0.2971	0.6312	0.1812	0.8209	0.6131	0.7192
Ens_MV	0.3532	0.7622	0.2382	0.6810	0.7714	0.7891
Ens_LR	0.6514	0.9321	0.7081	0.6033	0.9701	0.7894
Ens_NB	0.6522	0.9321	0.7083	0.6024	0.9701	0.7892
Ens_NN	0.6516	0.9321	0.7081	0.6034	0.9701	0.7891
Ens_DT	0.6516	0.9321	0.7081	0.6034	0.9701	0.7891
Ens_SVM	0.6521	0.9321	0.7083	0.6048	0.9701	0.7893

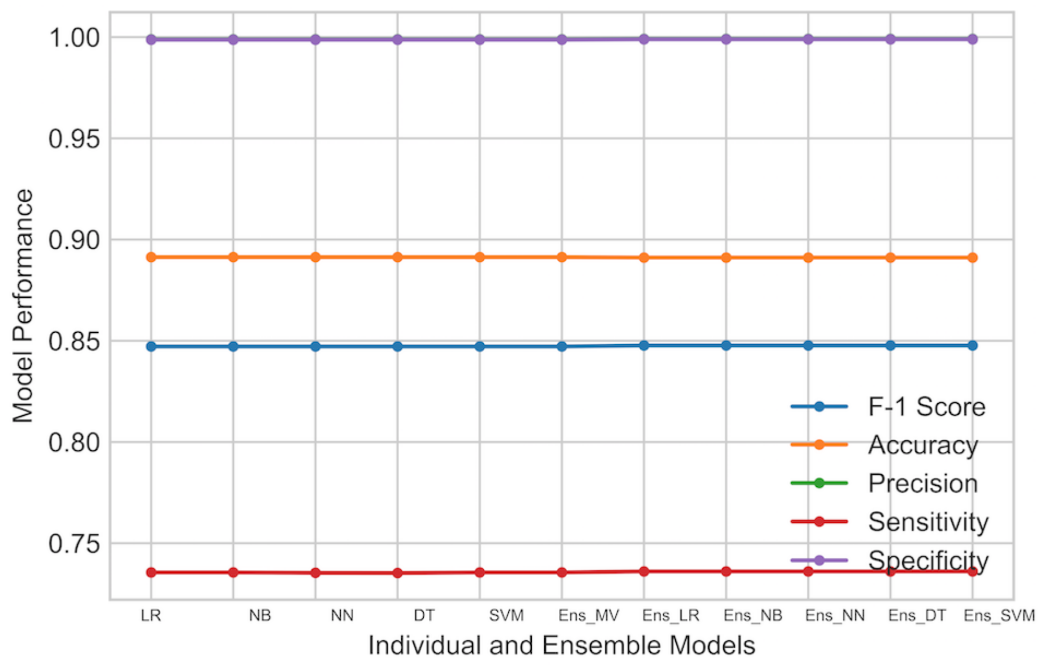


FIGURE A.5: Graph of the evaluation metrics values for supervised individual and ensemble models for the 14-bus-AM2 dataset.

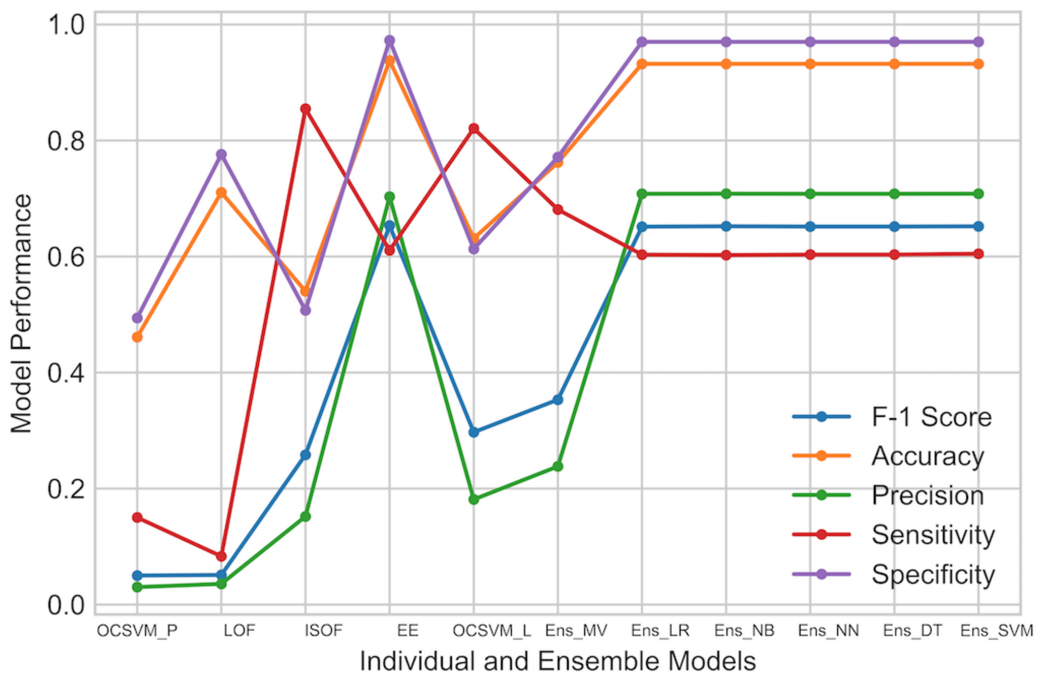


FIGURE A.6: Graph of the evaluation metrics values for unsupervised individual and ensemble models for the 14-bus-AM2 dataset.

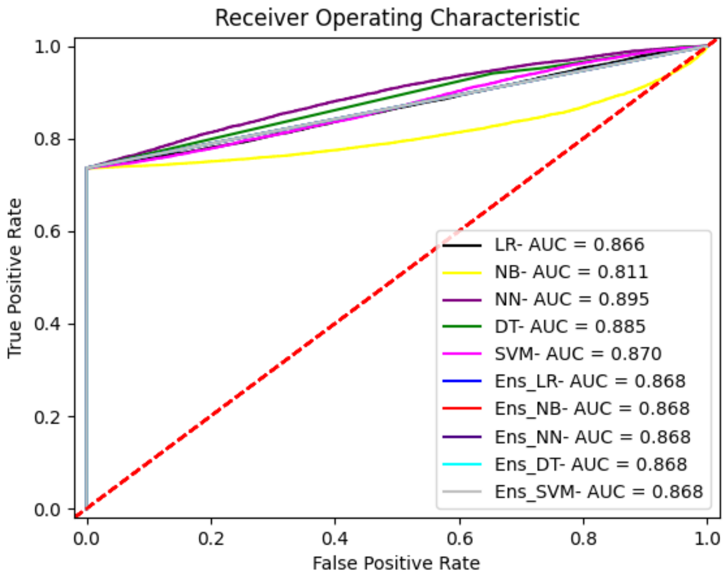


FIGURE A.7: ROC curves for the supervised individual and ensemble models for 14-bus-AM2 dataset.

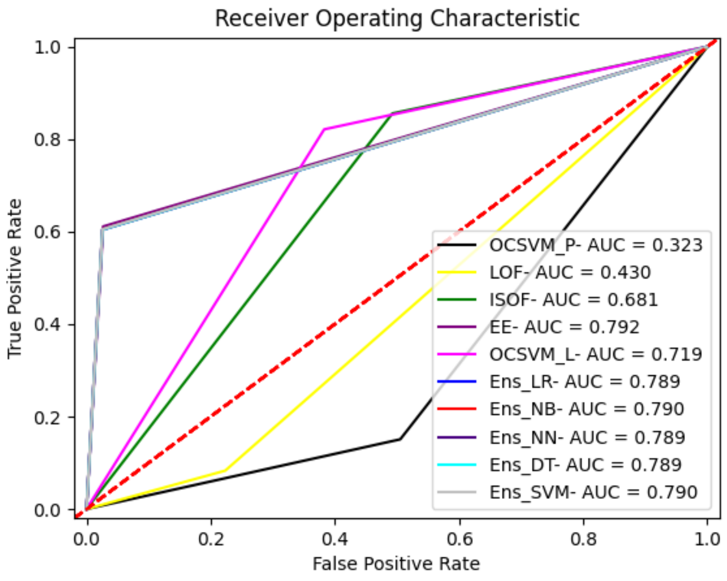


FIGURE A.8: ROC curves for the unsupervised individual and ensemble models for 14-bus-AM2 dataset.

The evaluation metric values for supervised and unsupervised individual and ensemble models for 14-bus-AM3 dataset in Tables A.5 and A.6. The corresponding bar-graphs are shown in Figures A.9 and A.10. The ROC curves are shown in Figures A.11 and A.12.

TABLE A.5: Evaluation metrics values for supervised individual and ensemble models using the 14-bus-AM3 dataset.

Models	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**ROC AUC**
LR	0.8479	0.8974	0.9994	0.7362	0.9998	0.8660
NB	0.8479	0.8974	0.9994	0.7362	0.9998	0.8111
NN	0.8479	0.8974	0.9994	0.7363	0.9998	0.8954
DT	0.8478	0.8974	0.9994	0.7361	0.9998	0.8846
SVM	0.8479	0.8974	0.9994	0.7362	0.9998	0.8698
Ens_MV	0.8479	0.8974	0.9994	0.7362	0.9998	0.8688
Ens_LR	0.8482	0.8964	0.9996	0.7367	0.9999	0.8682
Ens_NB	0.8482	0.8964	0.9996	0.7367	0.9999	0.8682
Ens_NN	0.8482	0.8964	0.9996	0.7367	0.9999	0.8682
Ens_DT	0.8482	0.8964	0.9996	0.7367	0.9999	0.8682
Ens_SVM	0.8482	0.8964	0.9996	0.7367	0.9999	0.8682

TABLE A.6: Evaluation metrics values for unsupervised individual and ensemble models using the 14-bus-AM3 dataset.

Models	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**ROC AUC**
OCSVM_P	0.0500	0.4623	0.03000	0.1508	0.4947	0.3227
LOF	0.0512	0.7114	0.0371	0.0830	0.7766	0.4298
ISOF	0.2589	0.5400	0.1525	0.8555	0.5078	0.6814
EE	0.6540	0.9392	0.7034	0.6110	0.9733	0.7921
OCSVM_L	0.2978	0.6363	0.1819	0.8211	0.6172	0.7191
Ens_MV	0.3538	0.7663	0.2390	0.6813	0.7752	0.7890
Ens_LR	0.6522	0.9395	0.7086	0.6042	0.9743	0.7892
Ens_NB	0.6529	0.9395	0.70893	0.6051	0.9743	0.7896
Ens_NN	0.6522	0.9395	0.7086	0.6042	0.9743	0.7892
Ens_DT	0.6522	0.9395	0.7086	0.6042	0.9743	0.7892
Ens_SVM	0.6529	0.9395	0.7089	0.6051	0.9743	0.7896

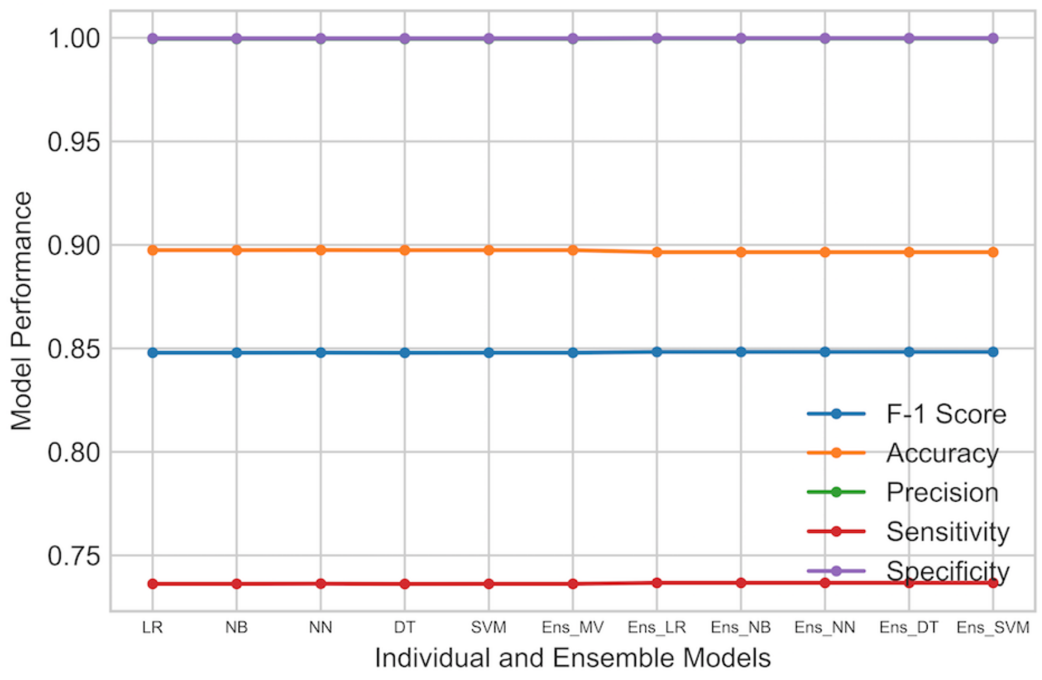


FIGURE A.9: Graph of the evaluation metrics values for supervised individual and ensemble models for the 14-bus-AM3 dataset.

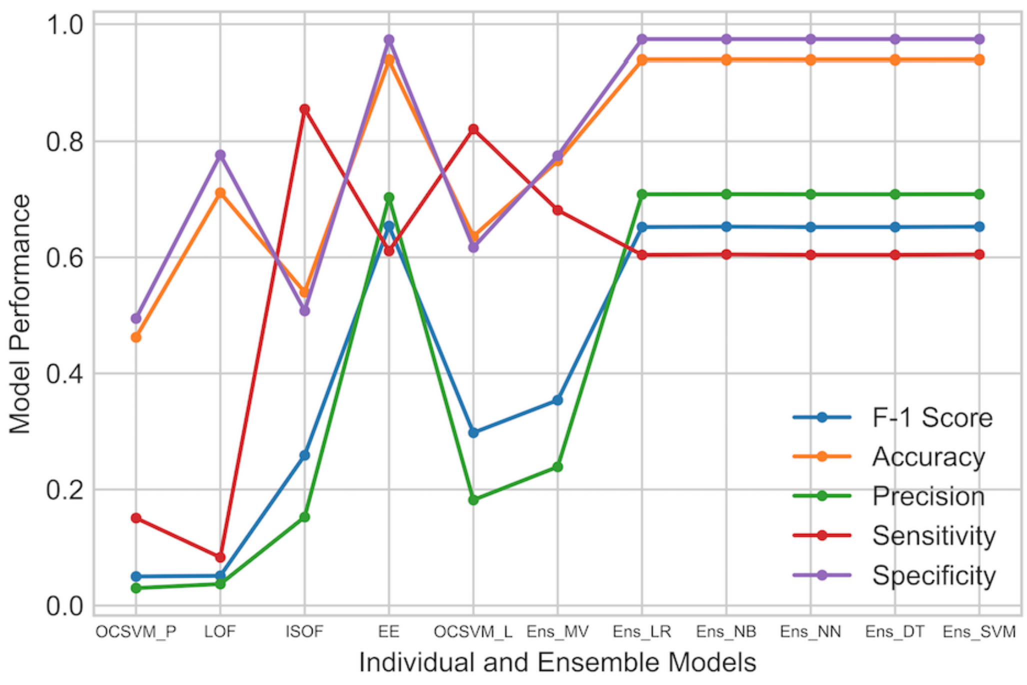


FIGURE A.10: Graph of the evaluation metrics values for unsupervised individual and ensemble models for the 14-bus-AM3 dataset.

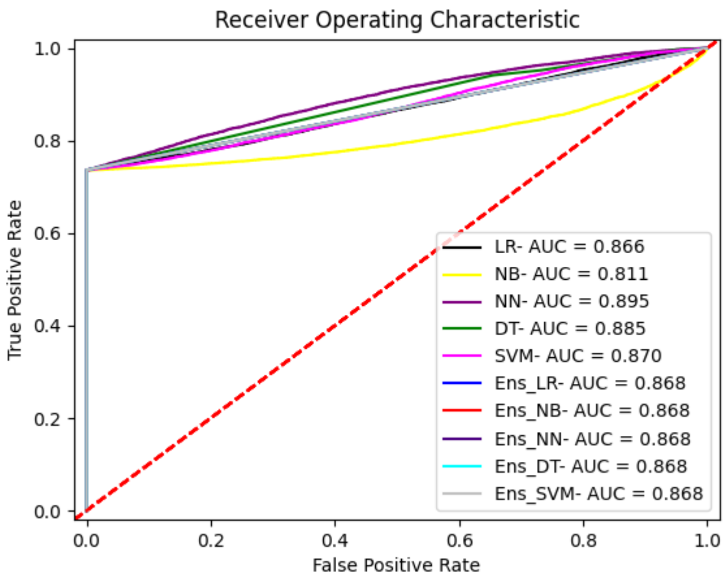


FIGURE A.11: ROC curves for the supervised individual and ensemble models for 14-bus-AM3 dataset.

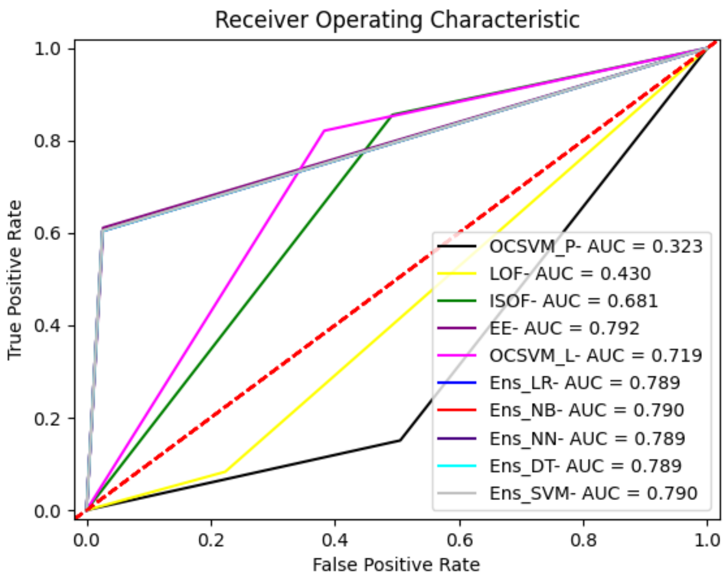


FIGURE A.12: ROC curves for the unsupervised individual and ensemble models for 14-bus-AM3 dataset.

The evaluation metric values for supervised and unsupervised individual and ensemble models for 57-bus-AM1 dataset in Tables A.7 and A.8. The corresponding bar-graphs are shown in Figures A.13 and A.14. The ROC curves are shown in Figures A.15 and A.16.

TABLE A.7: Evaluation metrics values for supervised individual and ensemble models using the 57-bus-AM1 dataset.

Models	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC
LR	0.8369	0.8826	0.9897	0.7288	0.9921	0.8559
NB	0.8369	0.8826	0.9897	0.7288	0.9921	0.8012
NN	0.8369	0.8826	0.9897	0.7288	0.9921	0.8591
DT	0.8368	0.8820	0.9897	0.7288	0.9921	0.8711
SVM	0.8369	0.8826	0.9897	0.7288	0.9921	0.8587
Ens_MV	0.8369	0.8826	0.9897	0.7304	0.9921	0.8591
Ens_LR	0.8424	0.8893	0.9903	0.7261	0.9921	0.8591
En_NB	0.8424	0.8893	0.9903	0.7261	0.9921	0.8591
Ens_NN	0.8424	0.8893	0.9903	0.7261	0.9921	0.8591
Ens_DT	0.8424	0.8893	0.9903	0.7261	0.9921	0.8591
Ens_SVM	0.8424	0.8893	0.9903	0.7261	0.9921	0.8591

TABLE A.8: Evaluation metrics values for unsupervised individual and ensemble models using the 57-bus-AM1 dataset.

Models	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC
OCSVM_P	0.0833	0.4531	0.0477	0.2609	0.4656	0.3702
LOF	0.1366	0.8439	0.1611	0.1231	0.9241	0.5283
ISOF	0.3719	0.7897	0.2575	0.6693	0.8012	0.7393
EE	0.6331	0.9189	0.5564	0.7198	0.9389	0.8327
OCSVM_L	0.1722	0.5972	0.1021	0.5583	0.4864	0.5277
Ens_MV	0.4366	0.8818	0.4107	0.4601	0.9309	-
Ens_LR	0.6412	0.9289	0.6697	0.6092	0.9691	0.7912
Ens_NB	0.5509	0.9011	0.4613	0.6574	0.9201	0.7969
Ens_NN	0.6202	0.9188	0.5375	0.7201	0.9486	0.8341
Ens_DT	0.6607	0.9387	0.6601	0.6477	0.9545	0.8114
Ens_SVM	0.6607	0.9387	0.6601	0.6477	0.9545	0.8114

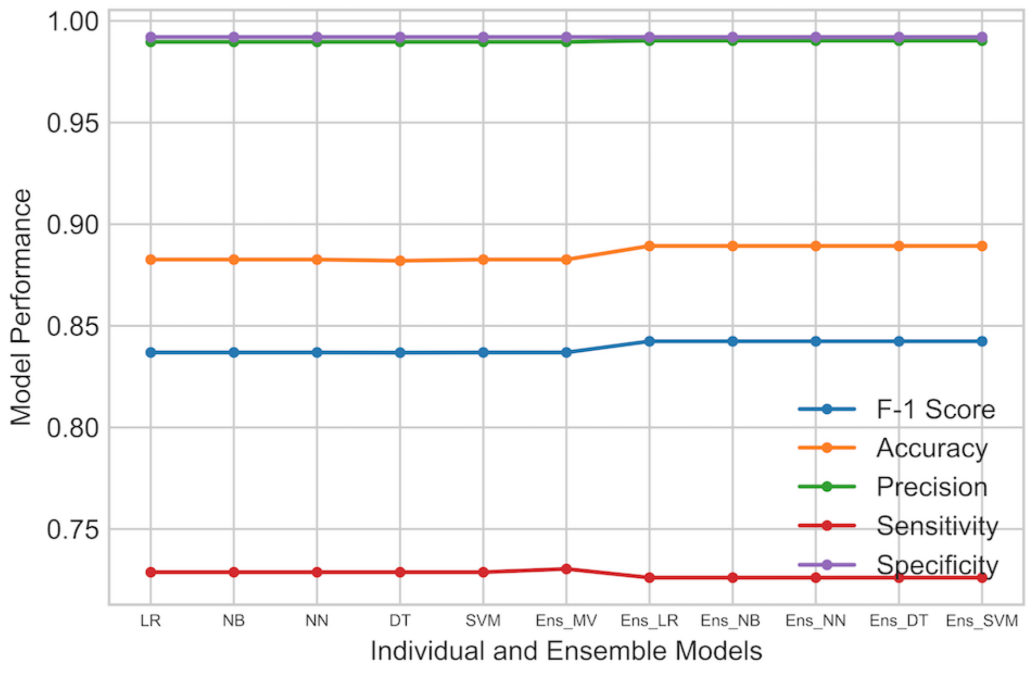


FIGURE A.13: Graph of the evaluation metrics values for supervised individual and ensemble models for the 57-bus-AM1 dataset.

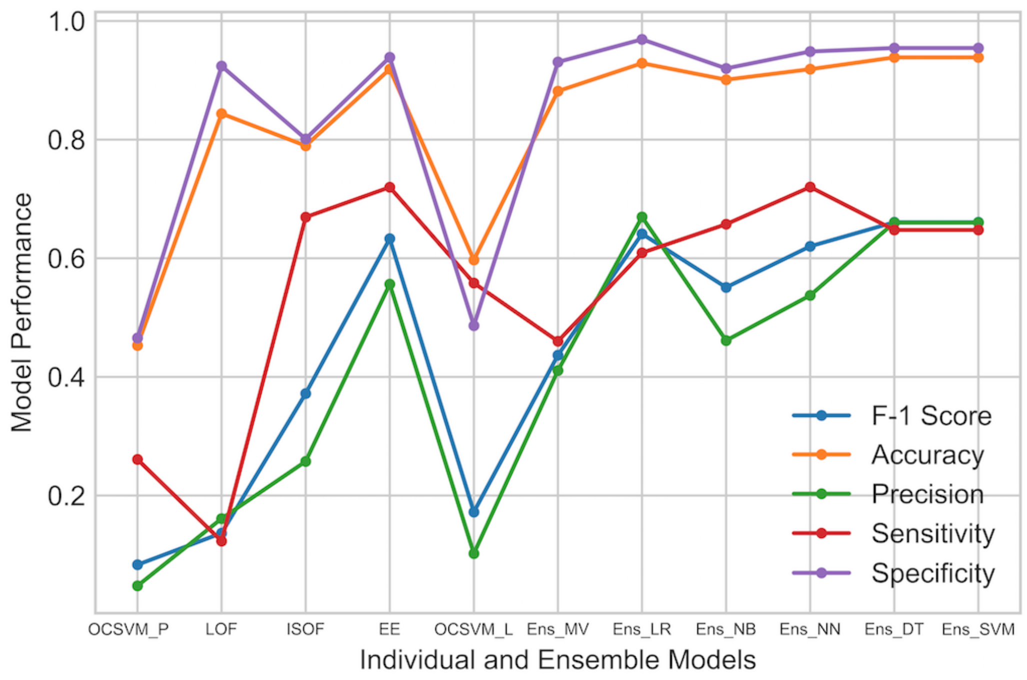


FIGURE A.14: Graph of the evaluation metrics values for unsupervised individual and ensemble models for the 57-bus-AM1 dataset.

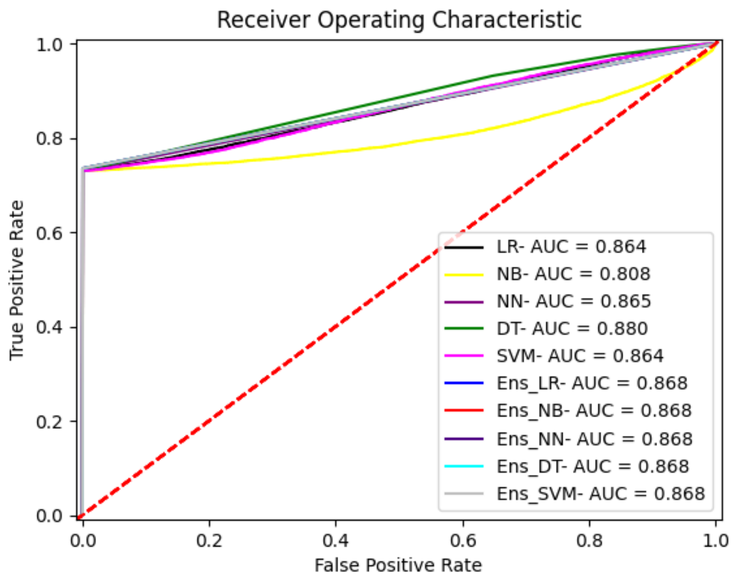


FIGURE A.15: ROC curves for the supervised individual and ensemble models for 57-bus-AM1 dataset.

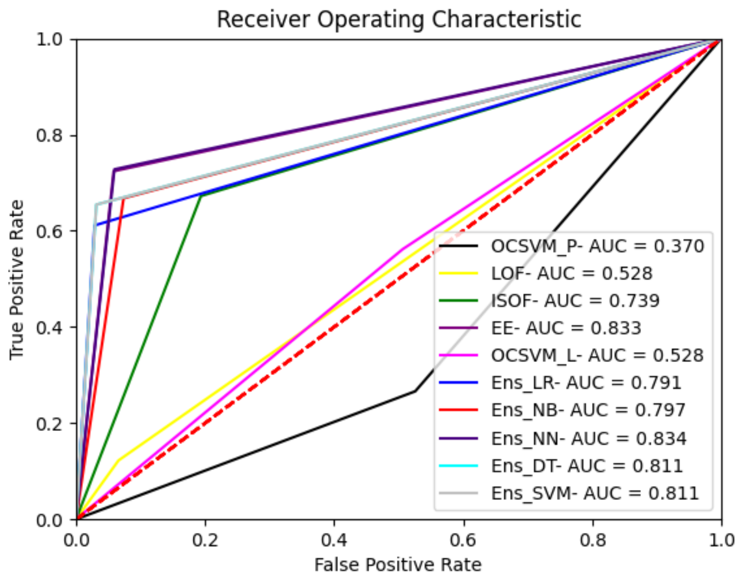


FIGURE A.16: ROC curves for the unsupervised individual and ensemble models for 57-bus-AM1 dataset.

The evaluation metric values for supervised and unsupervised individual and ensemble models for 57-bus-AM2 dataset in Tables A.9 and A.10. The corresponding bar-graphs are shown in Figures A.17 and A.18. The ROC curves are shown in Figures A.19 and A.20.

TABLE A.9: Evaluation metrics values for supervised individual and ensemble models using the 57-bus-AM2 dataset.

Models	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**ROC AUC**
LR	0.8451	0.8862	0.9991	0.7305	0.9994	0.8633
NB	0.8449	0.8862	0.9991	0.7305	0.9994	0.7871
NN	0.8449	0.8862	0.9991	0.7305	0.9994	0.9056
DT	0.8448	0.8862	0.9991	0.7305	0.9994	0.8835
SVM	0.8449	0.8862	0.9991	0.7305	0.9994	0.8671
Ens_MV	0.8449	0.8862	0.9991	0.7305	0.9994	0.8652
Ens_LR	0.8442	0.8845	0.9991	0.7302	0.9995	0.8652
Ens_NB	0.8442	0.8845	0.9991	0.7302	0.9995	0.8652
Ens_NN	0.8442	0.8845	0.9991	0.7302	0.9995	0.8652
Ens_DT	0.8442	0.8845	0.9991	0.7302	0.9995	0.8652
Ens_SVM	0.8442	0.8845	0.9991	0.7302	0.9995	0.8652

TABLE A.10: Evaluation metrics values for unsupervised individual and ensemble models using the 57-bus-AM2 dataset.

Models	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**ROC AUC**
OCSVM_P	0.0492	0.4601	0.0292	0.1504	0.4925	0.3211
LOF	0.0481	0.7102	0.0342	0.0777	0.7762	0.4271
ISOF	0.2533	0.5354	0.1491	0.8432	0.5041	0.6742
EE	0.6420	0.9388	0.7115	0.5841	0.9751	0.7803
OCSVM_L	0.0503	0.5531	0.0309	0.1256	0.5971	0.3617
Ens_MV	0.0525	0.7221	0.0382	0.0819	0.7861	0.7786
Ens_LR	0.6401	0.9393	0.7380	0.5651	0.9788	0.7723
Ens_NB	0.6457	0.9401	0.7353	0.5756	0.9788	0.7773
Ens_NN	0.6441	0.9400	0.7352	0.5731	0.9788	0.7753
Ens_DT	0.6447	0.9402	0.7356	0.5742	0.9788	0.7760
Ens_SVM	0.6457	0.9402	0.7353	0.5757	0.9788	0.7771

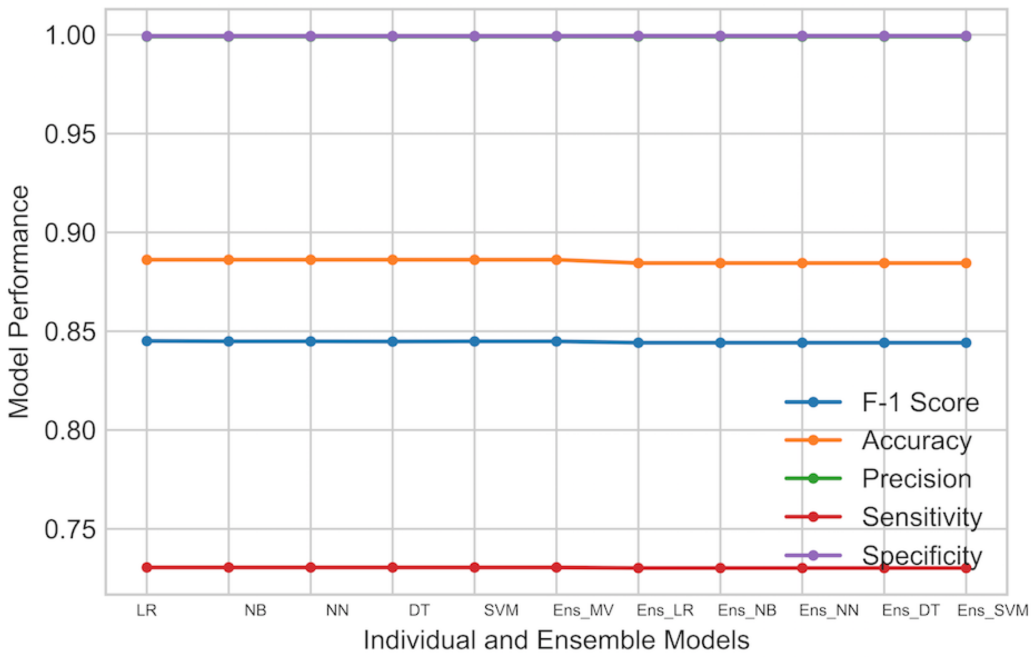


FIGURE A.17: Graph of the evaluation metrics values for supervised individual and ensemble models for the 57-bus-AM2 dataset.

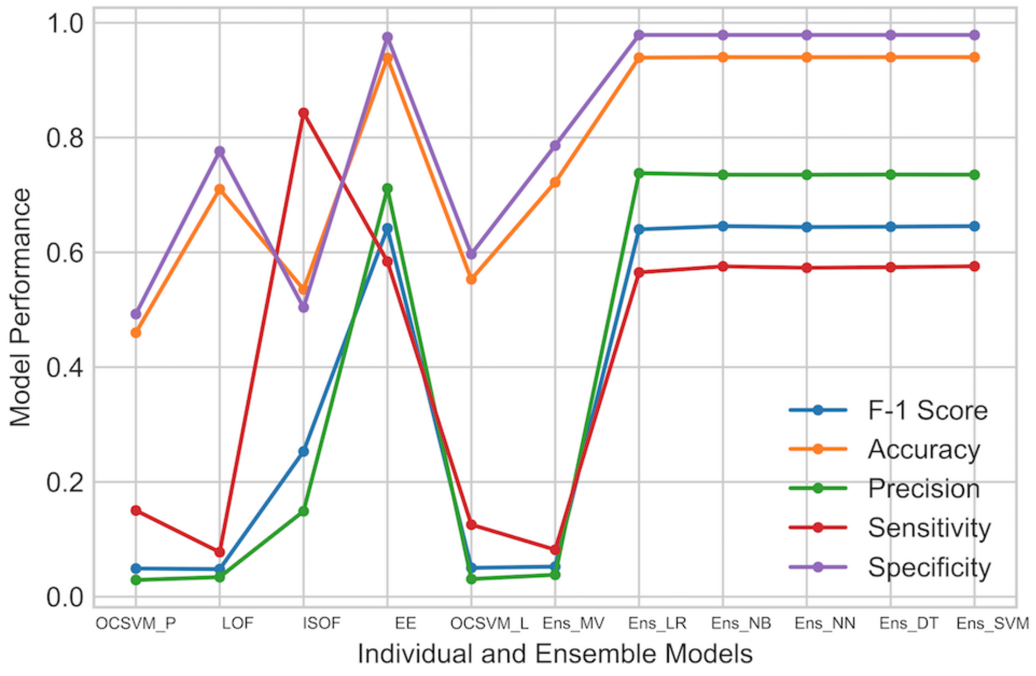


FIGURE A.18: Graph of the evaluation metrics values for unsupervised individual and ensemble models for the 57-bus-AM2 dataset.

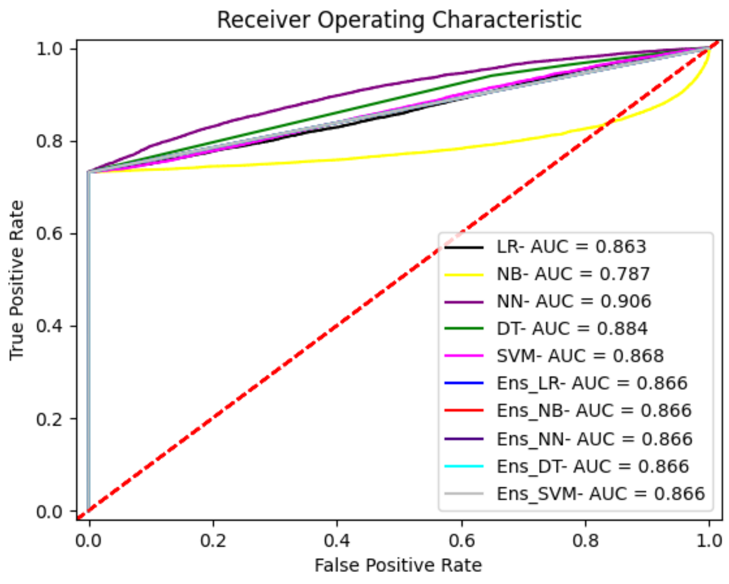


FIGURE A.19: ROC curves for the supervised individual and ensemble models for 57-bus-AM2 dataset.

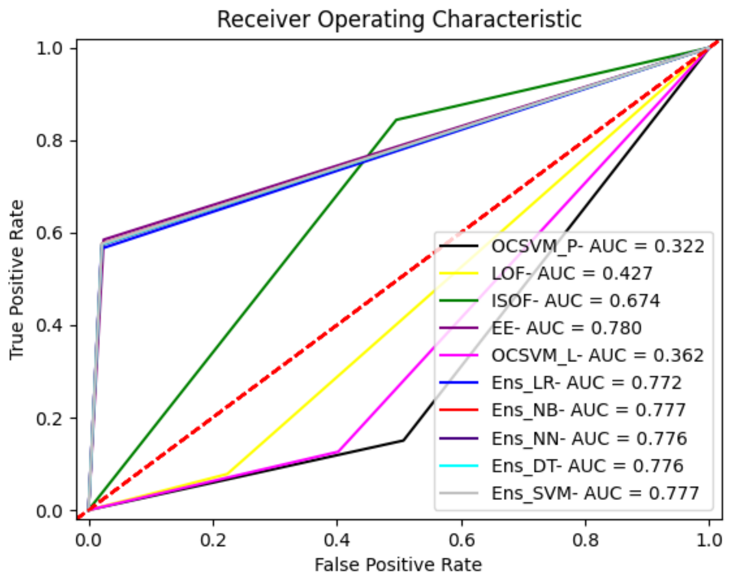


FIGURE A.20: ROC curves for the unsupervised individual and ensemble models for 57-bus-AM2 dataset.

The evaluation metric values for supervised and unsupervised individual and ensemble models for 14-bus-AM3 dataset in Tables A.11 and A.12. The corresponding bar-graphs are shown in Figures A.21 and A.22. The ROC curves are shown in Figures A.23 and A.24.

TABLE A.11: Evaluation metrics values for supervised individual and ensemble models using the 57-bus-AM3 dataset.

Models	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC
LR	0.8454	0.8866	0.9996	0.7322	0.9998	0.8632
NB	0.8453	0.8866	0.9996	0.7322	0.9998	0.7872
NN	0.8453	0.8866	0.9996	0.7322	0.9998	0.9059
DT	0.8452	0.8866	0.9996	0.7321	0.9998	0.8838
SVM	0.8453	0.8866	0.9996	0.7322	0.9998	0.8675
Ens_MV	0.8453	0.8866	0.9996	0.7322	0.9998	0.8655
Ens_LR	0.8446	0.8848	0.9996	0.7313	0.9999	0.8655
Ens_NB	0.8446	0.8848	0.9996	0.7313	0.9999	0.8655
Ens_NN	0.8446	0.8848	0.9996	0.7313	0.9999	0.8655
Ens_DT	0.8446	0.8848	0.9996	0.7313	0.9999	0.8655
Ens_SVM	0.8446	0.8848	0.9996	0.7313	0.9999	0.8655

TABLE A.12: Evaluation metrics values for unsupervised individual and ensemble models using the 57-bus-AM3 dataset.

Models	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC
OCSVM_P	0.0495	0.4606	0.0296	0.1503	0.4927	0.3215
LOF	0.0481	0.7110	0.0347	0.0780	0.7766	0.4272
ISOF	0.2538	0.5360	0.1494	0.8439	0.5043	0.6740
EE	0.6421	0.9390	0.7119	0.5848	0.9756	0.7802
OCSVM_L	0.0502	0.5535	0.0313	0.1261	0.5977	0.3619
Ens_MV	0.0528	0.7226	0.0387	0.0826	0.7867	0.7789
Ens_LR	0.6404	0.9398	0.7382	0.5655	0.9791	0.7722
Ens_NB	0.6462	0.9402	0.7359	0.5760	0.9791	0.7772
Ens_NN	0.6445	0.9400	0.7358	0.5734	0.9791	0.7759
Ens_DT	0.6452	0.9401	0.7361	0.5743	0.9791	0.7763
Ens_SVM	0.6462	0.9402	0.7359	0.5760	0.9791	0.7772

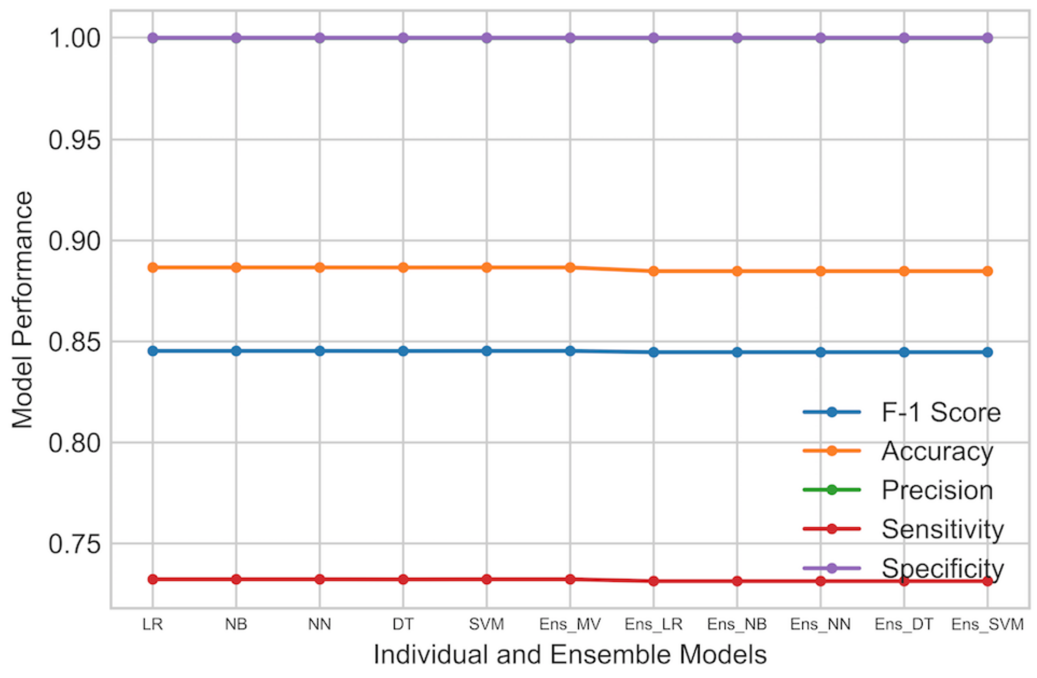


FIGURE A.21: Graph of the evaluation metrics values for supervised individual and ensemble models for the 57-bus-AM3 dataset.

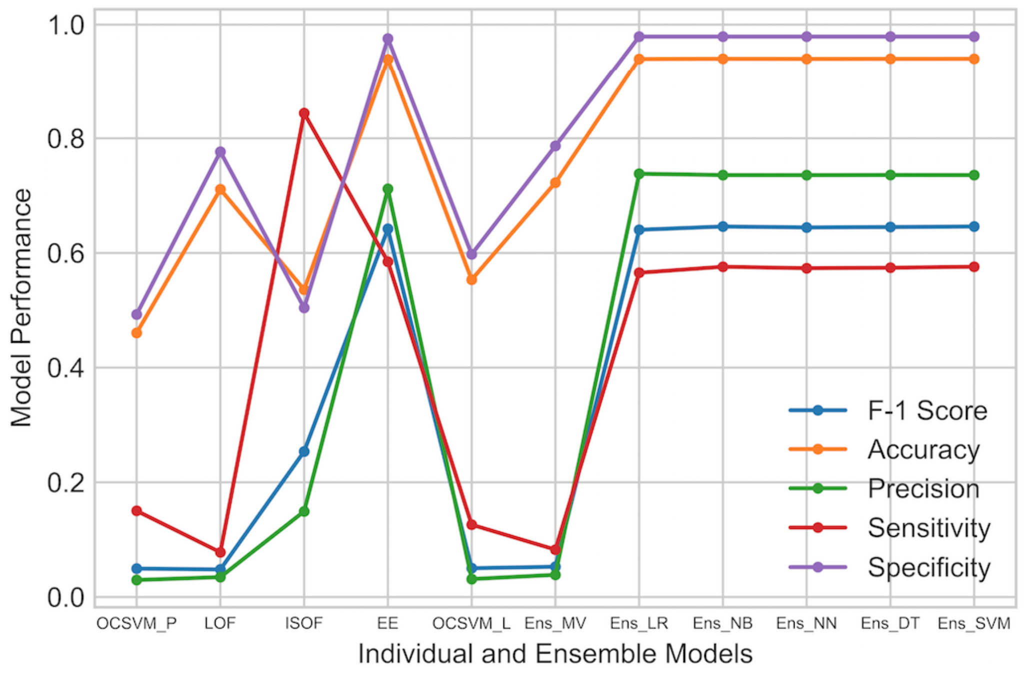


FIGURE A.22: Graph of the evaluation metrics values for unsupervised individual and ensemble models for the 57-bus-AM3 dataset.

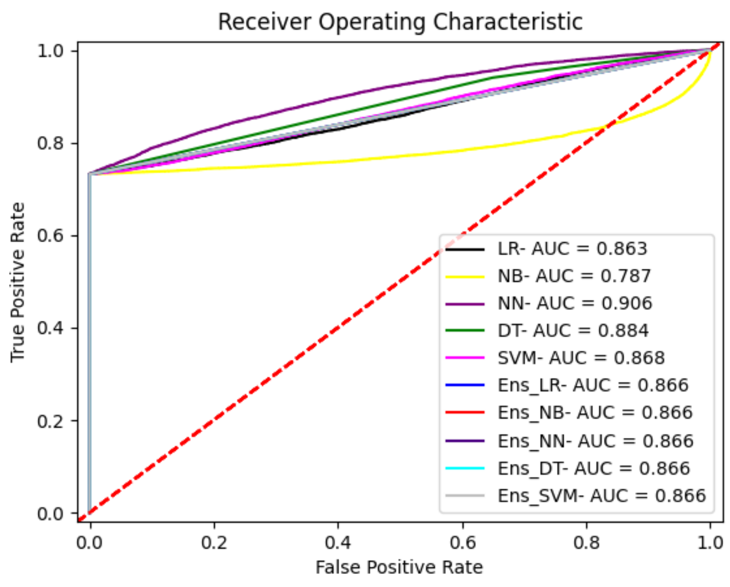


FIGURE A.23: ROC curves for the supervised individual and ensemble models for 57-bus-AM3 dataset

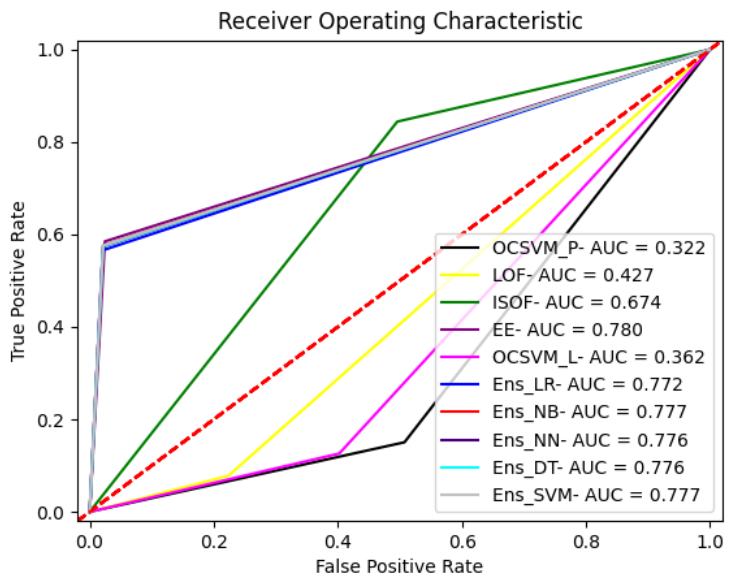


FIGURE A.24: ROC curves for the unsupervised individual and ensemble models for 57-bus-AM3 dataset

A.2 RESULTS OF ARTIFICIAL NEURAL NETWORK EXPERIMENTS

In this section, the performance metrics values for all six datasets from the experiments with artificial neural network methods from Chapter 8 are presented.

Table A.13 and corresponding graph in Figure A.25 show the performance metrics values for artificial neural network models with different hidden layers and different hidden units for the 14-Bus-AM1 dataset.

TABLE A.13: Evaluation metrics values for neural networks models with different hidden layers and different hidden units using the 14-bus-AM1 test dataset. The Time is elapsed time for training.

Layers	Units	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC	Time
2	50	0.8902	0.9120	0.9023	0.8784	0.9555	0.9674	32m 5s
	100	0.9282	0.9415	0.9176	0.9384	0.9565	0.9855	33m 11s
	150	0.9494	0.9588	0.9389	0.9595	0.9587	0.9863	36m 44s
	200	0.9528	0.9603	0.9414	0.9618	0.9613	0.9919	40m 32s
	250	0.9633	0.9667	0.9574	0.9667	0.9628	0.9924	44m 26s
	300	0.9596	0.9688	0.9484	0.9697	0.9773	0.9922	47m 13s
	400	0.9755	0.9717	0.9613	0.977	0.9792	0.9935	49m 40s
	450	0.9756	0.9724	0.9596	0.9775	0.9777	0.9938	42m 33s
	500	0.9683	0.9709	0.9441	0.9817	0.9626	0.9936	52m 2s
	550	0.9656	0.9733	0.954	0.9483	0.9866	0.9925	60m 55s
	600	0.9713	0.9737	0.9503	0.9814	0.9682	0.9949	1h 5m
	650	0.9715	0.9749	0.9557	0.9686	0.9781	0.9948	1h 35m
	700	0.978	0.9814	0.9648	0.9798	0.9807	0.9944	2h 10m
	800	0.9767	0.9788	0.9637	0.9758	0.9818	0.9933	3h 1m
	900	0.9829	0.9836	0.9727	0.9777	0.9885	0.9935	4h 38m
	1000	0.9831	0.9737	0.9553	0.9655	0.9797	0.9942	5h 22m
	1200	0.9726	0.9815	0.9656	0.9815	0.9808	0.9932	6h 10m
	1500	0.9798	0.9832	0.9691	0.9815	0.9835	0.9934	8h 27m
	1800	0.9783	0.9797	0.9637	0.9795	0.98	0.9935	10h 38m
	2000	0.9778	0.9794	0.9658	0.9748	0.9836	0.9938	13h 34m
	2500	0.9693	0.9682	0.9413	0.9746	0.9631	0.9933	22h 10m
4	50	0.9275	0.9346	0.8868	0.9337	0.9398	0.9871	38m 3s
	100	0.9579	0.9605	0.9292	0.9639	0.9649	0.9902	44m 10s
	200	0.9687	0.9725	0.9517	0.9684	0.9757	0.9926	53m 58s
	400	0.9787	0.9813	0.9691	0.9726	0.988	0.9947	1h 10m
	600	0.9781	0.9808	0.9647	0.9802	0.9811	0.9954	3h 32m
	1000	0.9795	0.9825	0.9725	0.9706	0.9915	0.9957	10h 25m
	1200	0.9834	0.9853	0.9744	0.9824	0.9878	0.9956	23h 50m
	1500	0.9786	0.9813	0.9671	0.9778	0.984	0.9952	1d 12h
6	600	0.9672	0.9704	0.9443	0.9791	0.9643	0.9943	1d 1h
	1000	0.9811	0.9831	0.9704	0.9812	0.9852	0.9949	1d 8h
	1200	0.9760	0.9792	0.9595	0.9842	0.9741	0.9950	1d 13h
	1500	0.9704	0.9808	0.9664	0.9793	0.9715	0.9945	1d 23h
	1800	0.9704	0.9781	0.9612	0.9721	0.9702	0.9913	2d 21h

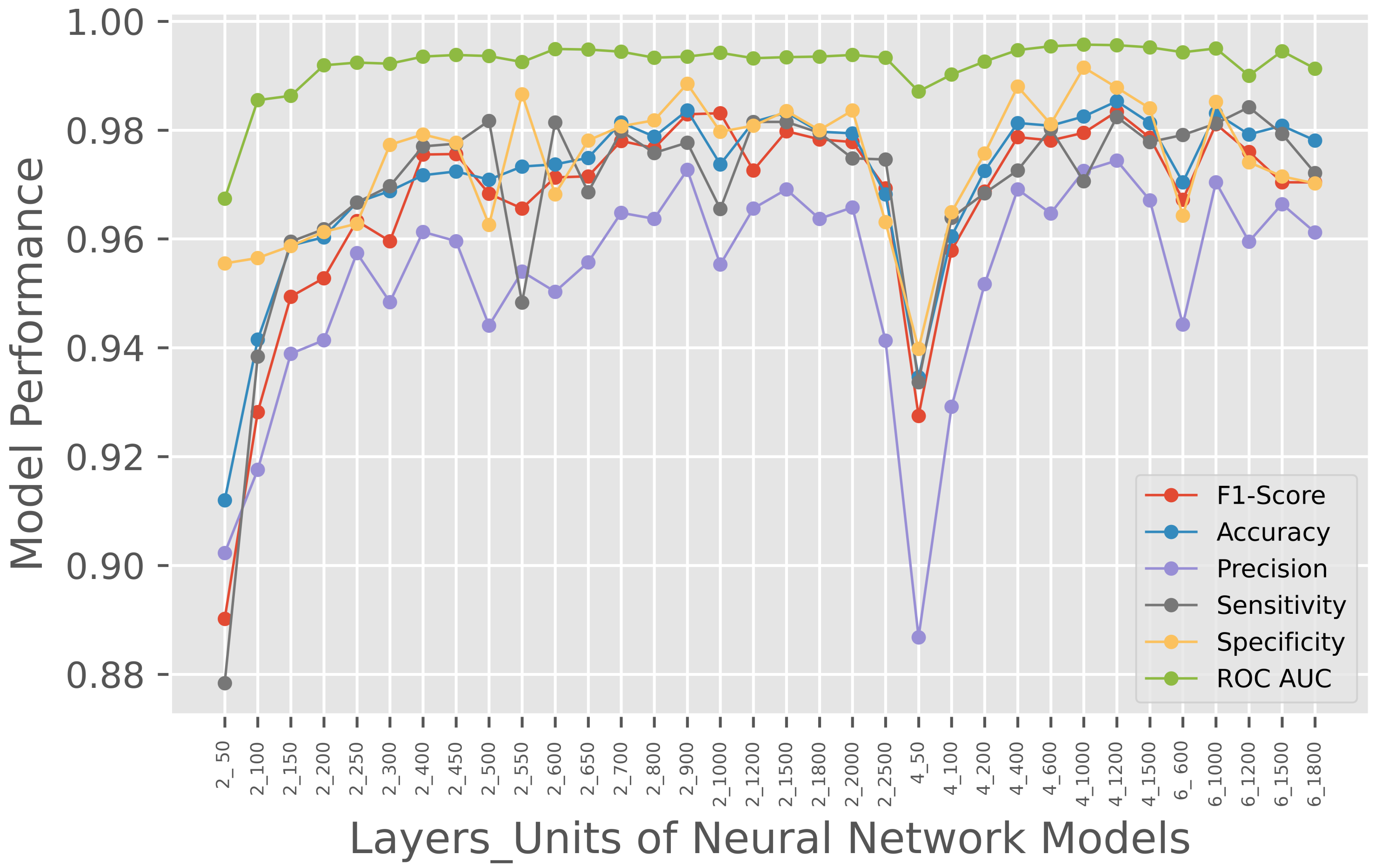


FIGURE A.25: Graph of metrics values for neural networks models with varying hidden layers and hidden units on 14-Bus-AM1 dataset.

Table A.14 and corresponding graph in Figure A.26 show the performance metrics values for artificial neural network models with different hidden layers and different hidden units for the 14-Bus-AM3 dataset.

TABLE A.14: Evaluation metrics values for neural networks models with different hidden layers and different hidden units using the 14-bus-AM2 test dataset. The Time is elapsed time for training.

Layers	Units	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC	Time
2	50	0.8900	0.9126	0.9024	0.8785	0.9559	0.9675	30m 44s
	100	0.9279	0.9414	0.9178	0.9385	0.9567	0.9857	31m 23s
	150	0.9491	0.9586	0.9388	0.9597	0.9588	0.9962	35m 20s
	200	0.9529	0.9604	0.9413	0.9617	0.9612	0.9917	33m 41s
	250	0.9634	0.9664	0.9575	0.9668	0.9629	0.9923	34m 56s
	300	0.9595	0.9685	0.9486	0.9699	0.9771	0.9928	40m 10s
	400	0.9753	0.9718	0.9612	0.9768	0.9791	0.9939	42m 43s
	450	0.9757	0.9726	0.9599	0.9778	0.9779	0.9939	47m 55s
	500	0.9681	0.9708	0.9443	0.9815	0.9628	0.9935	48m 39s
	550	0.9655	0.9731	0.9541	0.9481	0.9866	0.9928	52m 8s
	600	0.9711	0.9739	0.9501	0.9811	0.9680	0.9948	58m 43s
	650	0.9716	0.9747	0.9559	0.9687	0.9787	0.9945	1h 15m
	700	0.9781	0.9813	0.9646	0.9798	0.9809	0.9947	1h 43m
	800	0.9765	0.9788	0.9638	0.9757	0.9819	0.9931	2h 50m
	900	0.9827	0.9839	0.9729	0.9779	0.9884	0.9933	3h 41s
	1000	0.9832	0.9738	0.9554	0.9656	0.9797	0.9941	4h 50s
	1200	0.9728	0.9813	0.9659	0.9814	0.9807	0.9934	5h 53s
	1500	0.9799	0.9831	0.9689	0.9816	0.9834	0.9933	7h 6m
	1800	0.9781	0.9798	0.9635	0.9794	0.9801	0.9934	9h 34m
	2000	0.9779	0.9792	0.9657	0.9749	0.9837	0.9939	13h 28m
	2500	0.9692	0.9681	0.9410	0.9747	0.9633	0.9934	22h 4m
4	50	0.9276	0.9345	0.8862	0.9337	0.9398	0.9871	36m 33s
	100	0.9576	0.9603	0.9293	0.9639	0.9649	0.9902	40m 8s
	200	0.9685	0.9727	0.9509	0.9684	0.9757	0.9926	47m 56s
	400	0.9785	0.9814	0.9695	0.9726	0.988	0.9947	1h 9m
	600	0.9783	0.9811	0.9642	0.9802	0.9811	0.9954	3h 2m
	1000	0.9797	0.9822	0.9723	0.9706	0.9915	0.9957	9h 30m
	1200	0.9832	0.9855	0.9747	0.9824	0.9878	0.9956	22h 12m
	1500	0.9787	0.9810	0.9673	0.9778	0.984	0.9952	1d 13h
6	600	0.9670	0.9704	0.9442	0.9789	0.9643	0.9943	1d 0h
	1000	0.9812	0.9835	0.9711	0.9814	0.9854	0.9946	1d 7h
	1200	0.9761	0.9792	0.9593	0.9841	0.9742	0.9948	1d 14h
	1500	0.9714	0.9808	0.9662	0.9795	0.9715	0.9942	1d 21h
	1800	0.9708	0.9782	0.9612	0.9723	0.9701	0.9911	2d 12h

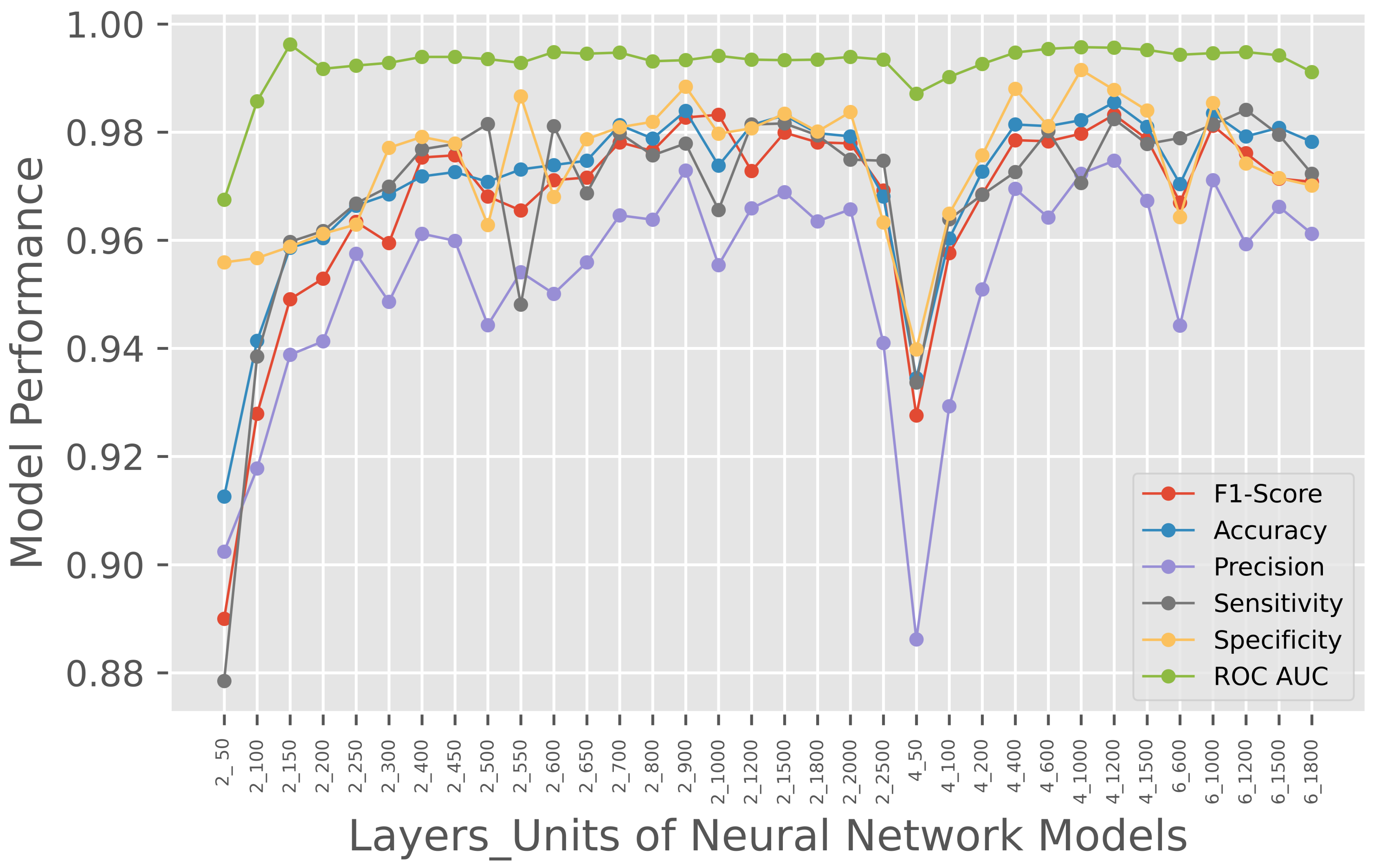


FIGURE A.26: Graph of metrics values for neural networks models with varying hidden layers and hidden units on 14-Bus-AM2 dataset.

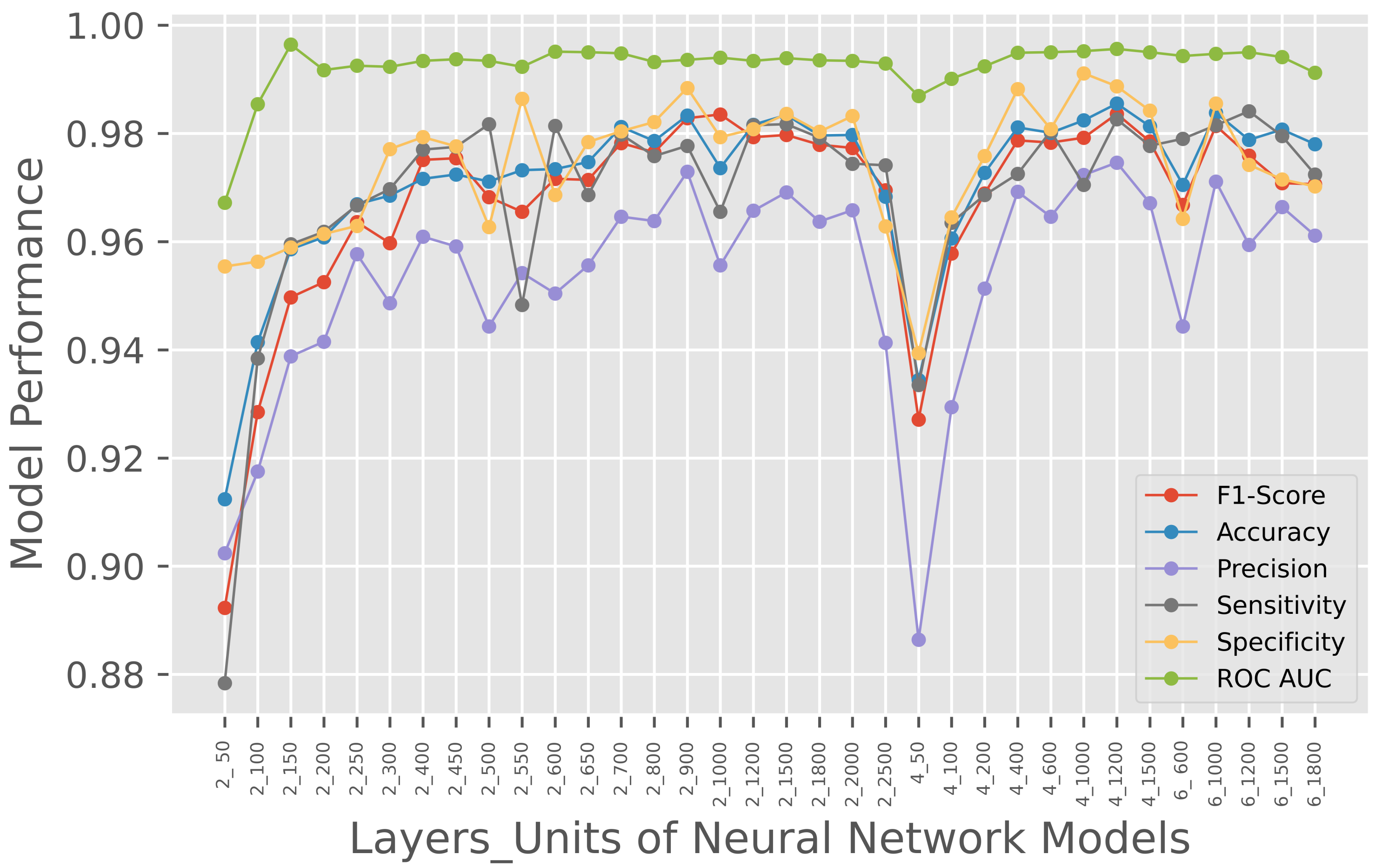


FIGURE A.27: Graph of metrics values for neural networks models with varying hidden layers and hidden units on 14-Bus-AM3 dataset.

Table A.15 and corresponding graph in Figure A.27 show the performance metrics values for artificial neural network models with different hidden layers and different hidden units for the 14-Bus-AM3 dataset.

TABLE A.15: Evaluation metrics values for neural networks models with different hidden layers and different hidden units using the 14-bus-AM3 test dataset. The Time is elapsed time for training.

Layers	Units	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC	Time
2	50	0.8923	0.9124	0.9024	0.8784	0.9554	0.9672	33m 1s
	100	0.9285	0.9414	0.9175	0.9384	0.9563	0.9854	30m 5s
	150	0.9497	0.9586	0.9388	0.9595	0.9589	0.9964	35m 6s
	200	0.9525	0.9608	0.9415	0.9618	0.9614	0.9917	45m 47s
	250	0.9636	0.9669	0.9577	0.9667	0.9629	0.9925	46m 11s
	300	0.9597	0.9685	0.9486	0.9697	0.9771	0.9923	47m 22s
	400	0.9751	0.9716	0.9609	0.977	0.9793	0.9934	42m 36s
	450	0.9754	0.9724	0.9591	0.9775	0.9776	0.9937	47m 33s
	500	0.9682	0.9711	0.9443	0.9817	0.9627	0.9934	52m 47s
	550	0.9655	0.9732	0.9542	0.9483	0.9864	0.9923	61m 29s
	600	0.9716	0.9734	0.9504	0.9814	0.9686	0.9951	68m 58s
	650	0.9714	0.9747	0.9556	0.9686	0.9784	0.995	79m 29s
	700	0.9782	0.9812	0.9646	0.9798	0.9804	0.9948	82m 55s
	800	0.9765	0.9786	0.9638	0.9758	0.9821	0.9932	1h 34m
	900	0.9828	0.9833	0.9729	0.9777	0.9884	0.9936	1h 52m
	1000	0.9835	0.9736	0.9556	0.9655	0.9793	0.994	2h 1m
	1200	0.9793	0.9816	0.9657	0.9815	0.9808	0.9934	3h 47m
	1500	0.9797	0.9834	0.9691	0.9817	0.9836	0.9939	4h 39m
	1800	0.9779	0.9796	0.9637	0.9792	0.9803	0.9935	10h 38m
	2000	0.9773	0.9797	0.9658	0.9744	0.9832	0.9934	14h 11m
	2500	0.9695	0.9683	0.9413	0.9741	0.9628	0.9929	24h 58m
4	50	0.9271	0.9344	0.8864	0.9335	0.9394	0.9869	21m 43s
	100	0.9578	0.9606	0.9294	0.9635	0.9645	0.9901	24m 49s
	200	0.9689	0.9727	0.9513	0.9686	0.9758	0.9924	33m 36s
	400	0.9787	0.9811	0.9692	0.9725	0.9882	0.9949	2h 2m
	600	0.9783	0.9801	0.9646	0.9804	0.9808	0.9950	1h 53m
	1000	0.9792	0.9824	0.9723	0.9705	0.9911	0.9952	14h 55m
	1200	0.9836	0.9855	0.9746	0.9826	0.9887	0.9956	23h 56m
	1500	0.9785	0.9813	0.9671	0.9777	0.9842	0.9950	1d 18h
6	600	0.9668	0.9705	0.9443	0.9790	0.9642	0.9943	23h 47m
	1000	0.9814	0.9838	0.9711	0.9815	0.9855	0.9947	1d 10h
	1200	0.9759	0.9788	0.9594	0.9841	0.9742	0.9950	1d 15h
	1500	0.9708	0.9807	0.9664	0.9795	0.9715	0.9941	2d 1h
	1800	0.9706	0.9780	0.9611	0.9724	0.9702	0.9912	2d 18h

Table A.16 and corresponding graph in Figure A.28 show the performance metrics values for artificial neural network models with different hidden layers and different hidden units for the 57-Bus-AM1 dataset.

TABLE A.16: Evaluation metrics values for neural networks models with different hidden layers and different hidden units using the 57-bus-AM1 test dataset. The Time is elapsed time for training.

Layers	Units	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC	Time
2	50	0.9203	0.9294	0.8793	0.9244	0.9332	0.9765	20m 33s
	100	0.9493	0.9553	0.9210	0.9553	0.9554	0.9861	21m 41s
	150	0.9415	0.9478	0.9253	0.9585	0.9523	0.9869	23m 22s
	200	0.9536	0.9588	0.9464	0.9606	0.9592	0.9897	24m 21s
	250	0.9615	0.9661	0.9569	0.9662	0.9626	0.9916	30m 2s
	300	0.9585	0.9633	0.9481	0.9694	0.9769	0.9914	46m 51s
	400	0.9746	0.9777	0.9605	0.9763	0.9787	0.9931	41m 36s
	450	0.9742	0.9771	0.9593	0.9769	0.9775	0.9932	48m 23s
	500	0.9668	0.9702	0.9435	0.9806	0.9625	0.9933	55m 1s
	550	0.9643	0.9692	0.9533	0.9472	0.9863	0.9923	1h 3m
	600	0.9701	0.9733	0.9494	0.9801	0.9677	0.9942	1h 11m
	650	0.9702	0.9737	0.9553	0.9680	0.9783	0.9931	1h 23m
	700	0.9771	0.9800	0.9645	0.9791	0.9806	0.9938	1h 25m
	800	0.9756	0.9786	0.9630	0.9747	0.9815	0.9924	1h 45m
	900	0.9807	0.9831	0.9722	0.9771	0.9880	0.9920	2h 3m
	1000	0.9689	0.973	0.9546	0.9646	0.9792	0.9939	2h 30m
	1200	0.9775	0.9805	0.9646	0.9804	0.9803	0.9924	4h 33m
	1500	0.9795	0.9822	0.9682	0.9810	0.9832	0.9926	5h 18m
	1800	0.9766	0.9792	0.9628	0.9782	0.9801	0.9922	11h 7m
	2000	0.9763	0.9793	0.9648	0.9741	0.9832	0.9931	14h 34m
	2500	0.9636	0.9679	0.9402	0.9742	0.9631	0.9930	1d 20m
4	50	0.9269	0.9338	0.8862	0.9331	0.9391	0.9863	24m 55s
	100	0.9574	0.9596	0.9287	0.9632	0.9642	0.9895	27m 29s
	200	0.9679	0.9718	0.9511	0.9677	0.9749	0.9921	36m 32s
	400	0.9778	0.9805	0.9685	0.9720	0.9872	0.9943	2h 50m
	600	0.9773	0.9801	0.9641	0.9796	0.9805	0.9949	5h 35m
	1000	0.9789	0.9817	0.9718	0.9701	0.9908	0.9951	16h 39m
	1200	0.9827	0.9847	0.9739	0.9819	0.9872	0.9951	1d 9h
	1500	0.9779	0.9808	0.9665	0.9772	0.9835	0.9945	1d 21h
6	600	0.9672	0.9709	0.9447	0.9791	0.9647	0.9947	22h 43s
	1000	0.9813	0.9838	0.9712	0.9821	0.9856	0.9949	1d 3h
	1200	0.9760	0.9788	0.9597	0.9841	0.9745	0.9949	1d 12h
	1500	0.9712	0.9811	0.9667	0.9799	0.9718	0.9947	1d 23h
	1800	0.9710	0.9778	0.9613	0.9726	0.9703	0.9912	2d 19h

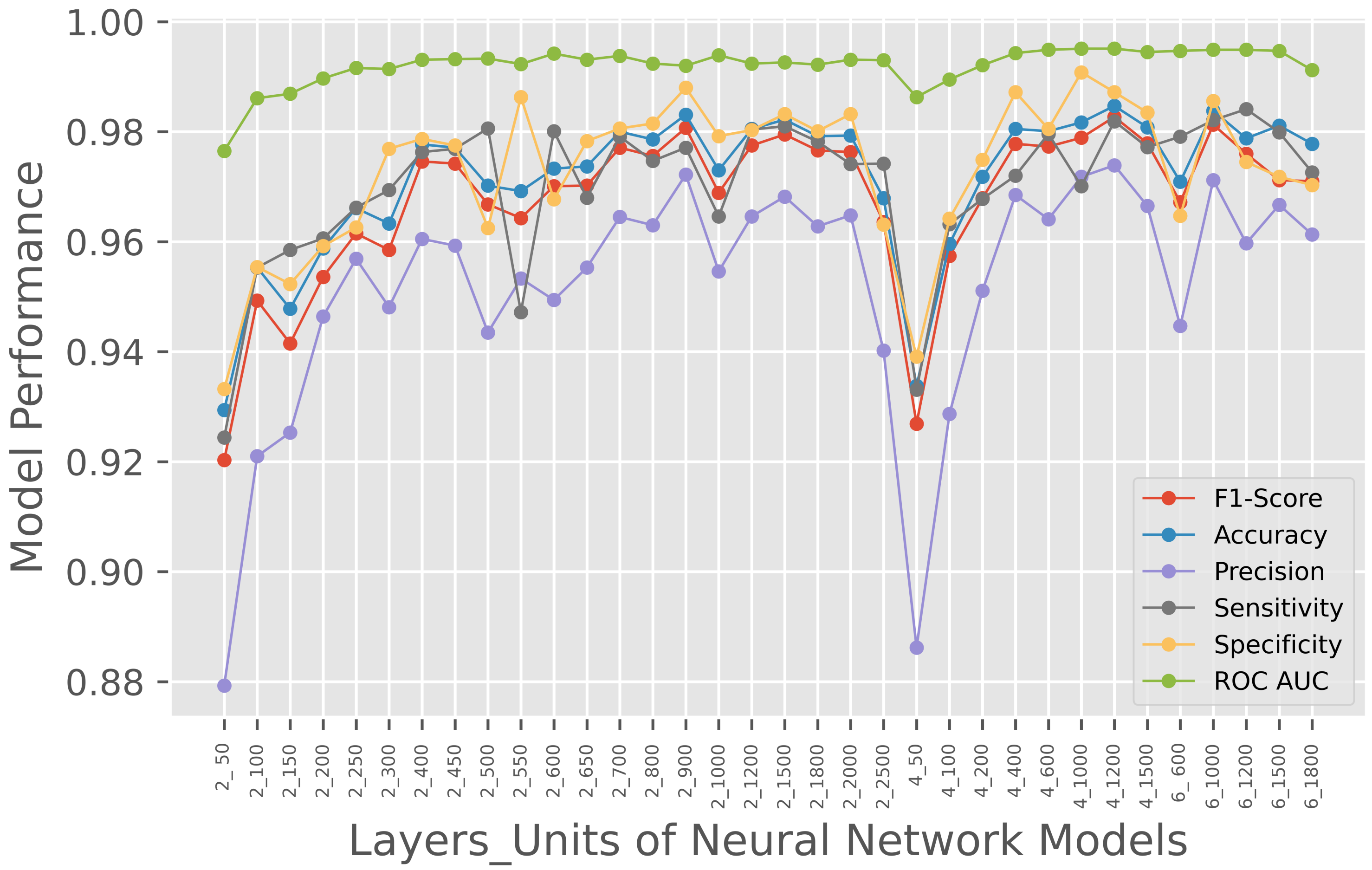


FIGURE A.28: Graph of metrics values for neural networks models with varying hidden layers and hidden units on 57-Bus-AM1 dataset.

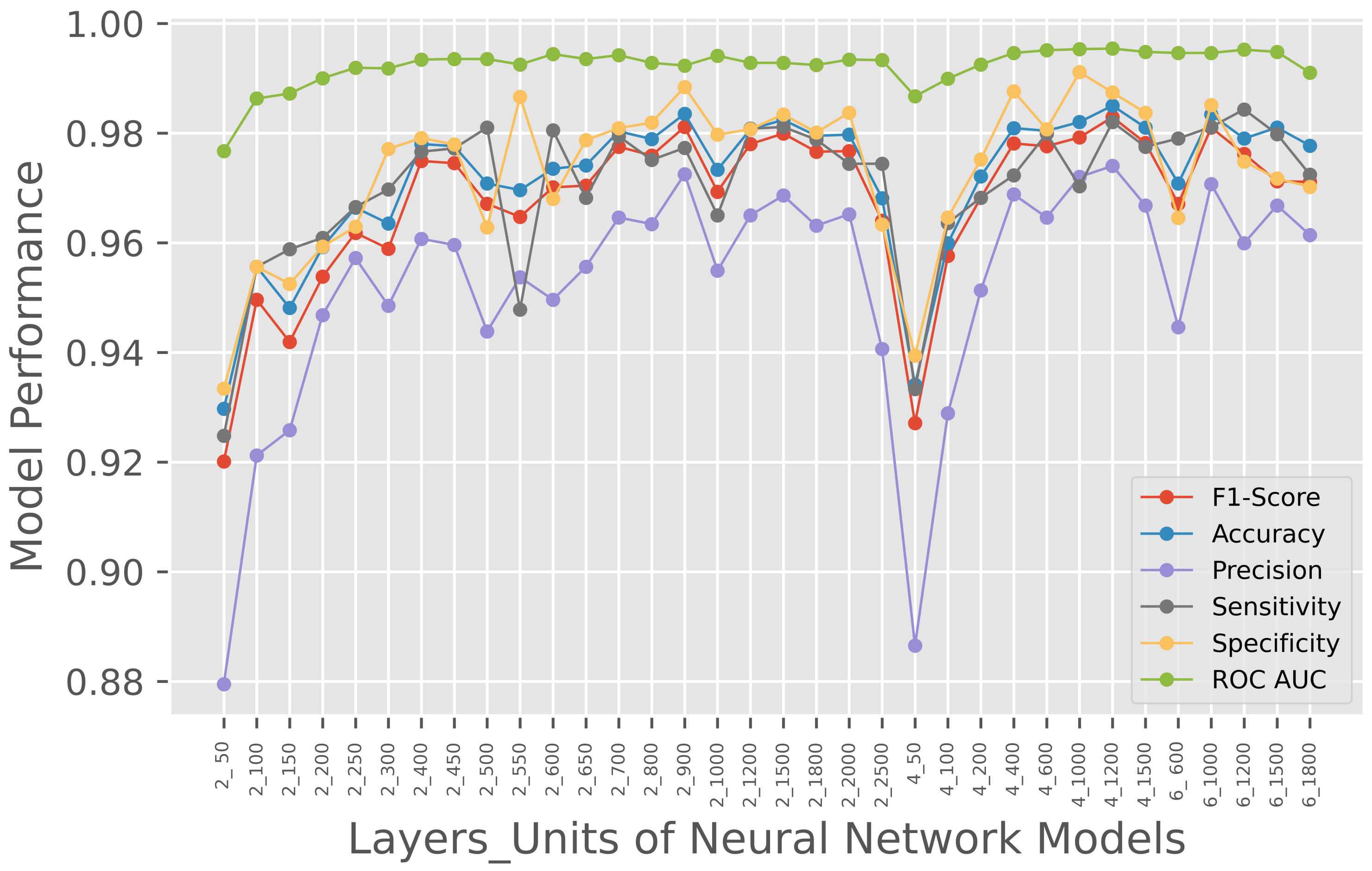


FIGURE A.29: Graph of metrics values for neural networks models with varying hidden layers and hidden units on 57-Bus-AM2 dataset.

Table A.17 and corresponding graph in Figure A.29 show the performance metrics values for artificial neural network models with different hidden layers and different hidden units for the 57-Bus-AM2 dataset.

TABLE A.17: Evaluation metrics values for neural networks models with different hidden layers and different hidden units using the 57-bus-AM2 test dataset. The Time is elapsed time for training.

Layers	Units	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC	Time
2	50	0.9201	0.9297	0.8795	0.9248	0.9334	0.9767	35m 40s
	100	0.9496	0.9556	0.9212	0.9556	0.9556	0.9863	38m 11s
	150	0.9419	0.9481	0.9258	0.9588	0.9525	0.9872	42m 20s
	200	0.9538	0.9592	0.9468	0.9609	0.9593	0.9900	46m 32s
	250	0.9618	0.9664	0.9572	0.9665	0.9629	0.9919	50m 12s
	300	0.9589	0.9635	0.9485	0.9697	0.9771	0.9918	53m 47s
	400	0.9749	0.9780	0.9607	0.9766	0.9791	0.9934	57m 21s
	450	0.9745	0.9776	0.9596	0.9772	0.9779	0.9935	1h 2m
	500	0.9671	0.9708	0.9438	0.9810	0.9628	0.9935	1h 8m
	550	0.9647	0.9696	0.9537	0.9478	0.9866	0.9925	1h 22m
	600	0.9701	0.9735	0.9496	0.9805	0.9680	0.9944	1h 36m
	650	0.9704	0.9741	0.9556	0.9682	0.9787	0.9935	2h 20m
	700	0.9775	0.9803	0.9646	0.9794	0.9809	0.9942	2h 58m
	800	0.9759	0.9789	0.9634	0.9751	0.9819	0.9928	3h 40m
	900	0.9811	0.9835	0.9725	0.9773	0.9884	0.9923	4h 27m
	1000	0.9693	0.9733	0.9549	0.9650	0.9797	0.9941	5h 10m
	1200	0.9780	0.9807	0.9650	0.9808	0.9807	0.9928	5h 57m
	1500	0.9799	0.9824	0.9686	0.9811	0.9834	0.9928	7h 27m
	1800	0.9766	0.9795	0.9631	0.9788	0.9801	0.9924	9h 5m
	2000	0.9767	0.9797	0.9652	0.9744	0.9837	0.9934	15h 20m
	2500	0.9640	0.9681	0.9406	0.9744	0.9633	0.9933	24h 8m
4	50	0.9271	0.9341	0.8865	0.9333	0.9394	0.9867	42m 3s
	100	0.9576	0.9599	0.9289	0.9636	0.9646	0.9899	48m 36s
	200	0.9682	0.9721	0.9513	0.9682	0.9752	0.9925	50m 48s
	400	0.9781	0.9809	0.9688	0.9723	0.9876	0.9946	1h 12m
	600	0.9776	0.9804	0.9646	0.9799	0.9807	0.9951	3h 54m
	1000	0.9792	0.9820	0.9720	0.9703	0.9911	0.9953	11h 43m
	1200	0.9829	0.9850	0.9740	0.9820	0.9874	0.9954	1d 3h
	1500	0.9782	0.9810	0.9668	0.9775	0.9837	0.9948	2d 4h
6	600	0.9671	0.9708	0.9446	0.979	0.9645	0.9946	1d 20h
	1000	0.9810	0.9834	0.9707	0.9811	0.9851	0.9946	2d 8h
	1200	0.9762	0.9790	0.9599	0.9843	0.9748	0.9952	2d 22h
	1500	0.9712	0.9810	0.9668	0.9798	0.9717	0.9948	3d 8h
	1800	0.9711	0.9777	0.9614	0.9724	0.9702	0.9910	3d 21h

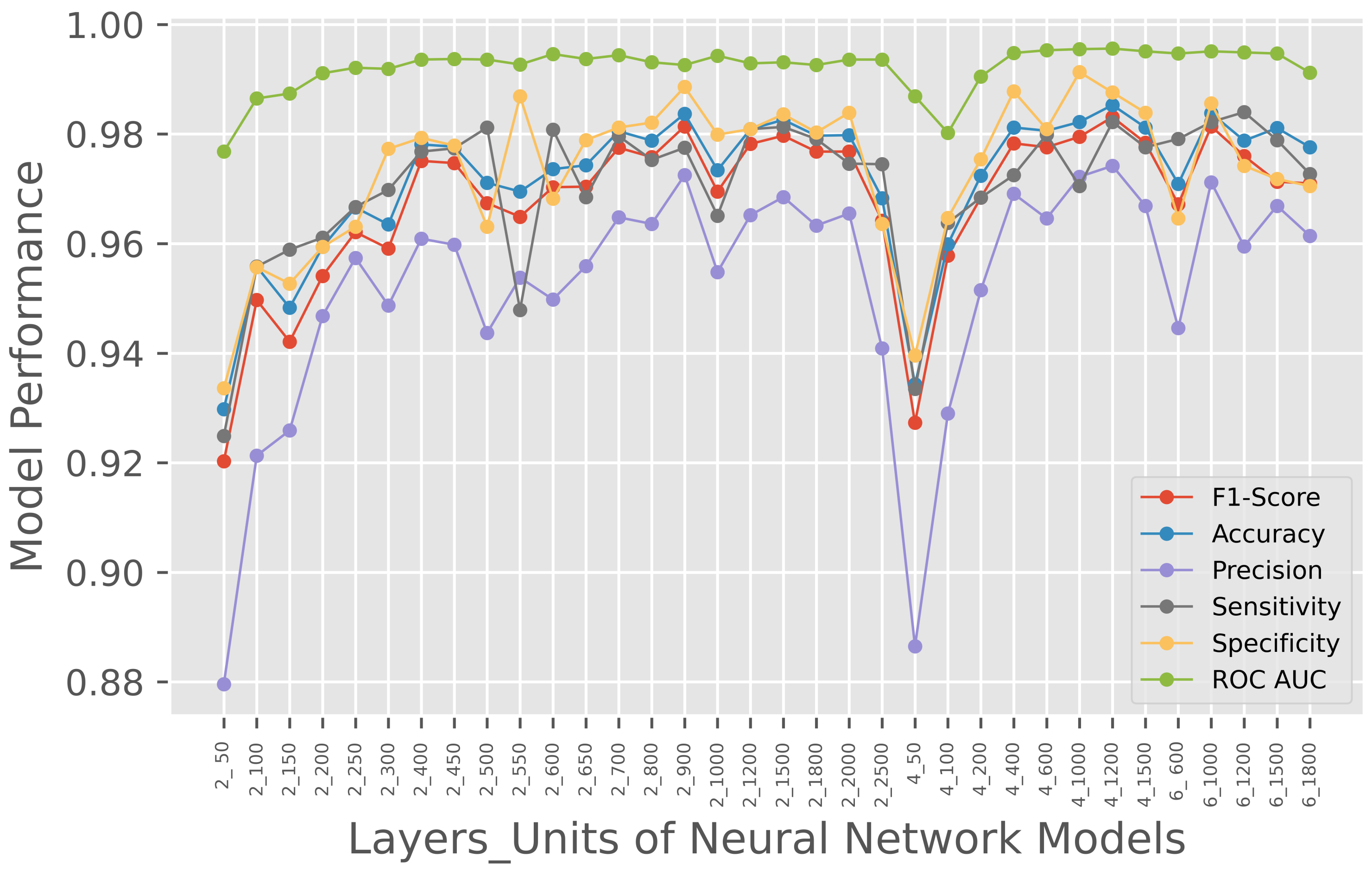


FIGURE A.30: Graph of metrics values for neural networks models with varying hidden layers and hidden units on 57-Bus-AM3 dataset.

Table A.18 and corresponding graph in Figure A.30 show the performance metrics values for artificial neural network models with different hidden layers and different hidden units for the 57-Bus-AM3 dataset.

TABLE A.18: Evaluation metrics values for neural networks models with different hidden layers and different hidden units using the 57-bus-AM3 test dataset. The Time is elapsed time for training.

Layers	Units	F1-Score	Accuracy	Precision	Sensitivity	Specificity	ROC AUC	Time
2	50	0.9203	0.9298	0.8796	0.9249	0.9336	0.9768	20m 32s
	100	0.9497	0.9558	0.9213	0.9558	0.9557	0.9865	22m 29s
	150	0.9421	0.9483	0.9259	0.9589	0.9527	0.9874	23m 52s
	200	0.9541	0.9594	0.9468	0.9611	0.9594	0.9911	25m 5s
	250	0.9621	0.9666	0.9574	0.9667	0.9631	0.9921	31m 15s
	300	0.9591	0.9635	0.9487	0.9698	0.9773	0.9919	48m 50s
	400	0.9751	0.9781	0.9609	0.9768	0.9793	0.9936	43m 7s
	450	0.9747	0.9777	0.9598	0.9774	0.9779	0.9937	49m 53s
	500	0.9674	0.9711	0.9437	0.9812	0.9631	0.9936	56m 38s
	550	0.9649	0.9695	0.9538	0.9479	0.9869	0.9927	1h 4m
	600	0.9703	0.9736	0.9498	0.9808	0.9682	0.9946	1h 12m
	650	0.9704	0.9743	0.9559	0.9685	0.9789	0.9937	1h 25m
	700	0.9775	0.9805	0.9648	0.9796	0.9812	0.9944	1h 26m
	800	0.9758	0.9788	0.9636	0.9753	0.9821	0.9931	1h 46m
	900	0.9814	0.9837	0.9725	0.9775	0.9886	0.9926	2h 4m
	1000	0.9695	0.9734	0.9548	0.9651	0.9799	0.9943	2h 47m
	1200	0.9782	0.9809	0.9652	0.9809	0.9809	0.9929	4h 35m
	1500	0.9797	0.9826	0.9685	0.9813	0.9836	0.9931	5h 33m
	1800	0.9768	0.9797	0.9633	0.9791	0.9803	0.9926	11h 28m
	2000	0.9768	0.9798	0.9655	0.9746	0.9839	0.9936	15h 9m
	2500	0.9642	0.9683	0.9409	0.9745	0.9636	0.9936	1d 4h
4	50	0.9273	0.9343	0.8865	0.9335	0.9396	0.9869	25m 30s
	100	0.9578	0.9599	0.929	0.9638	0.9647	0.9802	28m 18s
	200	0.9684	0.9724	0.9515	0.9684	0.9754	0.9905	37m 32s
	400	0.9783	0.9812	0.9691	0.9725	0.9878	0.9948	2h 52m
	600	0.9776	0.9806	0.9646	0.9799	0.9809	0.9953	5h 37m
	1000	0.9795	0.9822	0.9722	0.9705	0.9913	0.9955	16h 40m
	1200	0.9831	0.9853	0.9742	0.9822	0.9876	0.9956	1d 6h
	1500	0.9784	0.9812	0.9669	0.9776	0.9839	0.9951	1d 23h
6	600	0.9672	0.9709	0.9446	0.9791	0.9646	0.9947	23h 52m
	1000	0.9814	0.9838	0.9712	0.9822	0.9856	0.9951	1d 9h
	1200	0.9760	0.9788	0.9595	0.9840	0.9742	0.9949	1d 15h
	1500	0.9713	0.9811	0.9669	0.9789	0.9718	0.9947	2d 0h
	1800	0.9710	0.9776	0.9614	0.9727	0.9705	0.9912	2d 23h

A.3 RESULTS OF ELLIPTIC ENVELOPE EXPERIMENTS

In this section, the performance metrics values for all six datasets from the experiments with elliptic envelope method in Chapter 9 are presented.

In this section, the results from the model selection process in terms of the evaluation metrics are presented.

Table A.19 and corresponding graph in Figure A.31 show the effect of contamination rates for elliptic envelope models for the 14-Bus-AM1 dataset.

TABLE A.19: Performance metrics values for elliptic envelope models with different contamination rates using the 14-bus-AM1 test dataset. The elapsed time is for training.

Contamination Rate	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**Elapsed Time**
0.001	0.0619	0.9079	0.7531	0.0323	0.9989	44.12s
0.005	0.2038	0.9133	0.7483	0.1180	0.9959	45.19s
0.01	0.3467	0.9187	0.7122	0.2291	0.9904	41.09s
0.02	0.5627	0.9312	0.7000	0.4704	0.9791	40.37s
0.025	0.6386	0.9372	0.6960	0.5899	0.9732	45.92s
0.03	0.7183	0.9432	0.7286	0.7279	0.9787	46.94s
0.035	0.6996	0.9412	0.6742	0.7270	0.9635	42.88s
0.04	0.6868	0.9376	0.6504	0.7275	0.9594	43.55s
0.05	0.6605	0.9291	0.6011	0.7328	0.9495	45.16s
0.1	0.5466	0.8835	0.4313	0.7460	0.8978	44.37s
0.2	0.4190	0.7976	0.2870	0.7757	0.7998	43.48s
0.3	0.3435	0.7098	0.2182	0.8069	0.6998	44.22s
0.4	0.2930	0.6201	0.1776	0.8365	0.5976	41.33s
0.5	0.2571	0.5307	0.1511	0.8630	0.4962	45.68s

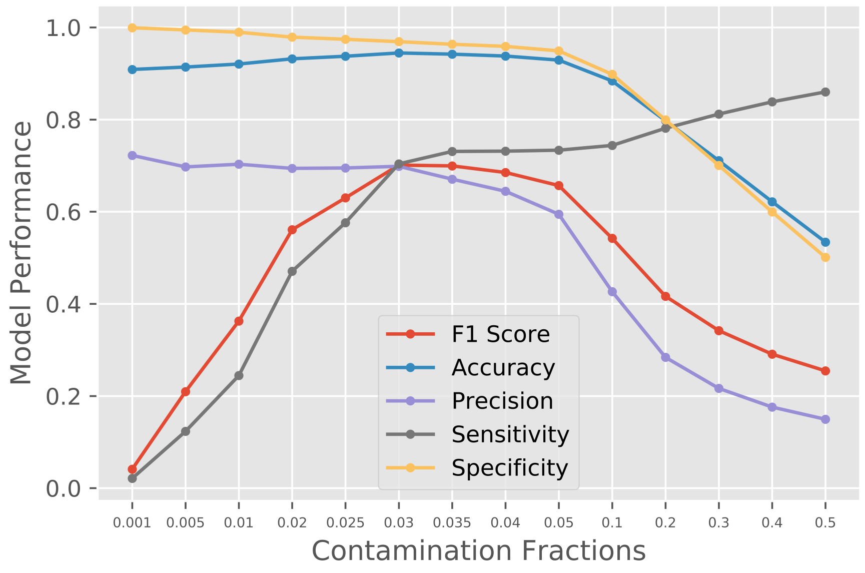


FIGURE A.31: Plot showing effect of contamination rate on model performance for 14-Bus-AM1 dataset.

Table A.20 and corresponding graph in Figure A.32 show the effect of contamination rates for elliptic envelope models for the 14-Bus-AM2 dataset.

TABLE A.20: Performance metrics values for elliptic envelope models with different contamination rates using the 14-bus-AM2 test dataset. The elapsed time is for training.

Contamination Rate	F1-Score	Accuracy	Precision	Sensitivity	Specificity	Elapsed Time
0.001	0.0452	0.9107	0.7636	0.0233	0.9993	38.04s
0.005	0.1931	0.9154	0.7204	0.1115	0.9957	40.32s
0.01	0.3311	0.9205	0.6995	0.2169	0.9907	39.88s
0.02	0.5435	0.9323	0.7011	0.4437	0.9811	40.38s
0.025	0.6151	0.9374	0.6956	0.5513	0.9759	39.24s
0.03	0.6823	0.9435	0.6969	0.6683	0.9710	40.05s
0.035	0.6945	0.9433	0.6792	0.7105	0.9665	39.77s
0.04	0.6787	0.9388	0.6486	0.7116	0.9615	41.93s
0.05	0.6525	0.9308	0.5994	0.7160	0.9522	41.48s
0.1	0.5356	0.8852	0.4232	0.7293	0.9008	42.62s
0.2	0.4083	0.8001	0.2792	0.7598	0.8041	40.76s
0.3	0.3310	0.7100	0.2094	0.7903	0.7020	43.05s
0.4	0.2825	0.6204	0.1705	0.8231	0.6001	44.48s
0.5	0.2503	0.5349	0.1466	0.8552	0.5029	41.25s

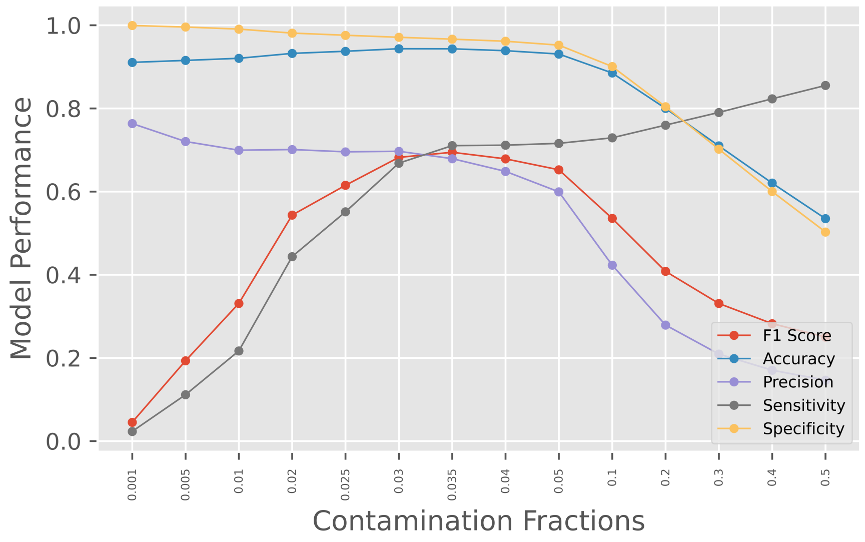


FIGURE A.32: Plot showing effect of contamination rate on model performance for 14-Bus-AM2 dataset.

Table A.21 and corresponding graph in Figure A.33 show the effect of contamination rates for elliptic envelope models for the 14-Bus-AM3 dataset.

TABLE A.21: Performance metrics values for elliptic envelope models with different contamination rates using the 14-bus-AM3 test dataset. The elapsed time is for training.

Contamination Rate	F1-Score	Accuracy	Precision	Sensitivity	Specificity	Elapsed Time
0.001	0.0459	0.9051	0.6389	0.0238	0.9986	39.50s
0.005	0.2031	0.9104	0.6886	0.1191	0.9943	41.16s
0.01	0.3411	0.9162	0.6926	0.2263	0.9893	42.16s
0.02	0.5362	0.9275	0.6935	0.4371	0.9795	40.60s
0.025	0.6122	0.9335	0.6945	0.5474	0.9745	42.91s
0.03	0.6873	0.9412	0.7013	0.6737	0.9696	41.22s
0.035	0.6882	0.9391	0.6762	0.7007	0.9644	42.63s
0.04	0.6723	0.9345	0.6457	0.7012	0.9592	39.05s
0.05	0.6472	0.9263	0.5982	0.7048	0.9498	41.64s
0.1	0.5409	0.8828	0.4331	0.7204	0.9000	40.48s
0.2	0.4159	0.7975	0.2874	0.7519	0.8023	37.53s
0.3	0.3431	0.7122	0.2196	0.7840	0.7045	43.09s
0.4	0.2961	0.6271	0.1808	0.8182	0.6069	39.98s
0.5	0.2598	0.5368	0.1534	0.8477	0.5038	39.90s

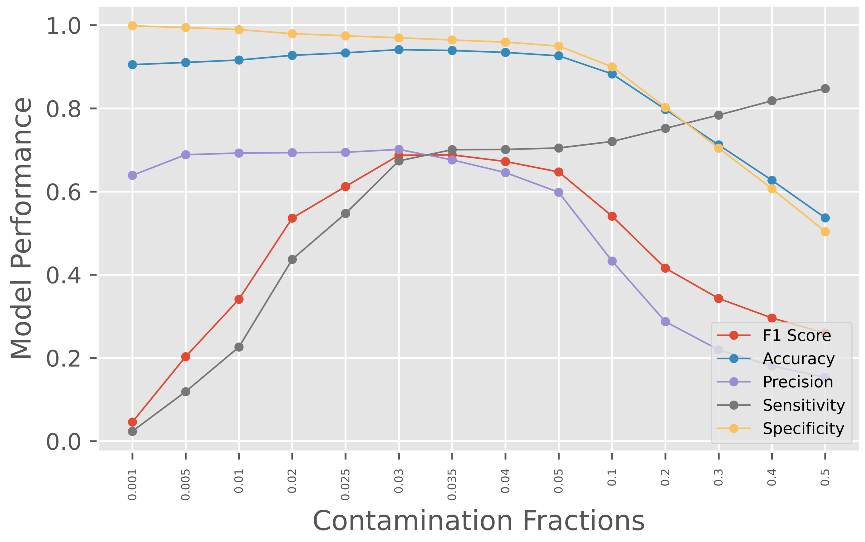


FIGURE A.33: Plot showing effect of contamination rate on model performance for 14-Bus-AM3 dataset.

Table A.22 and corresponding graph in Figure A.34 show the effect of contamination rates for elliptic envelope models for the 57-Bus-AM1 dataset.

TABLE A.22: Performance metrics values for elliptic envelope models with different contamination rates using the 57-bus-AM1 test dataset. The elapsed time is for training.

Contamination Rate	F1-Score	Accuracy	Precision	Sensitivity	Specificity	Elapsed Time
0.001	0.0673	0.9125	0.8077	0.0351	0.9992	56.60s
0.005	0.2134	0.9169	0.7166	0.1253	0.9951	57.87s
0.01	0.3569	0.9220	0.6901	0.2407	0.9893	57.64s
0.02	0.5476	0.9322	0.6848	0.4563	0.9792	55.38s
0.025	0.6257	0.9381	0.6855	0.5755	0.9739	58.44s
0.03	0.6863	0.9434	0.6846	0.6880	0.9687	55.33s
0.035	0.6920	0.9418	0.6606	0.7265	0.9631	56.94s
0.04	0.6791	0.9380	0.6350	0.7298	0.9585	54.89s
0.05	0.6494	0.9289	0.5835	0.7320	0.9484	53.82s
0.1	0.5341	0.8828	0.4158	0.7465	0.8963	57.64s
0.2	0.4042	0.7948	0.2735	0.7738	0.7969	53.72s
0.3	0.3309	0.7074	0.2083	0.8045	0.6978	54.64s
0.4	0.2812	0.6176	0.1692	0.8318	0.5964	56.08s
0.5	0.2473	0.5297	0.1445	0.8591	0.4972	55.42s

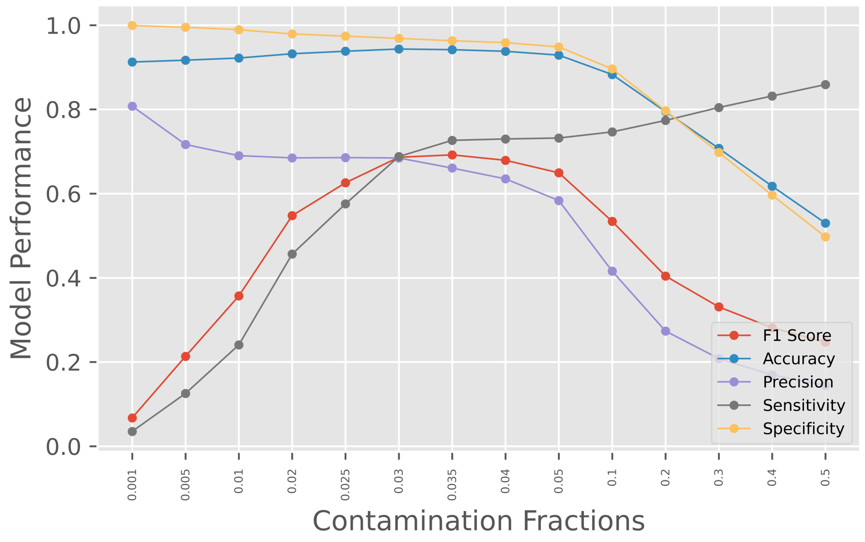


FIGURE A.34: Plot showing effect of contamination rate on model performance for 57-Bus-AM1 dataset.

Table A.23 and corresponding graph in Figure A.35 show the effect of contamination rates for elliptic envelope models for the 57-Bus-AM2 dataset.

TABLE A.23: Performance metrics values for elliptic envelope models with different contamination rates using the 57-bus-AM2 test dataset. The elapsed time is for training.

Contamination Rate	**F1-Score**	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**Elapsed Time**
0.001	0.0460	0.9080	0.6769	0.0238	0.9988	55.35s
0.005	0.1757	0.9114	0.6608	0.1013	0.9947	53.63s
0.01	0.3234	0.9175	0.6836	0.2118	0.9899	54.64s
0.02	0.5362	0.9293	0.6894	0.4388	0.9797	53.01s
0.025	0.6176	0.9357	0.6925	0.5574	0.9746	54.27s
0.03	0.6990	0.9443	0.7042	0.6939	0.9701	55.08s
0.035	0.7046	0.9432	0.6831	0.7275	0.9653	52.92s
0.04	0.6912	0.9394	0.6574	0.7286	0.9610	55.61s
0.05	0.6622	0.9306	0.6058	0.7302	0.9512	57.78s
0.1	0.5549	0.8877	0.4399	0.7514	0.9017	53.44s
0.2	0.4235	0.8019	0.2905	0.7811	0.8040	54.70s
0.3	0.3449	0.7135	0.2191	0.8099	0.7036	54.84s
0.4	0.2966	0.6264	0.1798	0.8456	0.6039	56.46s
0.5	0.2592	0.5369	0.1523	0.8700	0.5027	52.33s

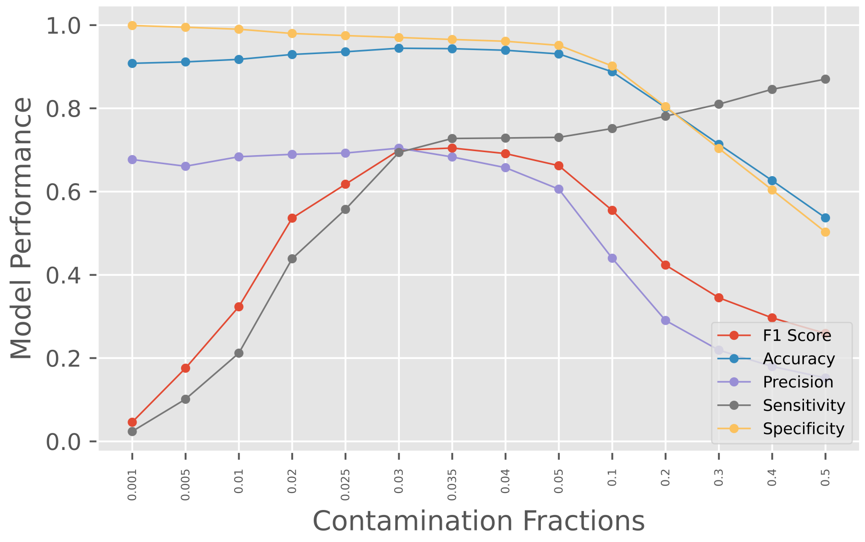


FIGURE A.35: Plot showing effect of contamination rate on model performance for 57-Bus-AM2 dataset.

Table A.24 and corresponding graph in Figure A.36 show the effect of contamination rates for elliptic envelope models for the 57-Bus-AM3 dataset.

TABLE A.24: Performance metrics values for elliptic envelope models with different contamination rates using the 57-bus-AM3 test dataset. The elapsed time is for training.

Contamination Rate	F1-Score	Accuracy	Precision	Sensitivity	Specificity	Elapsed Time
0.001	0.0581	0.9096	0.7179	0.0303	0.9988	57.95s
0.005	0.2036	0.9143	0.7051	0.1190	0.9950	60.37s
0.01	0.3422	0.9200	0.7037	0.2261	0.9903	53.62s
0.02	0.5555	0.9320	0.6980	0.4613	0.9798	56.36s
0.025	0.6216	0.9367	0.6914	0.5646	0.9744	57.33s
0.03	0.6832	0.9419	0.6865	0.6798	0.9685	56.45s
0.035	0.6903	0.9408	0.6663	0.7161	0.9636	56.16s
0.04	0.6729	0.9357	0.6334	0.7177	0.9579	53.98s
0.05	0.6451	0.9267	0.5823	0.7231	0.9474	54.07s
0.1	0.5433	0.8851	0.4285	0.7420	0.8996	53.64s
0.2	0.4147	0.7991	0.2834	0.7729	0.8017	54.35s
0.3	0.3401	0.7120	0.2155	0.8058	0.7025	53.25s
0.4	0.2903	0.6237	0.1757	0.8356	0.6022	52.03s
0.5	0.2533	0.5318	0.1484	0.8621	0.4983	52.63s

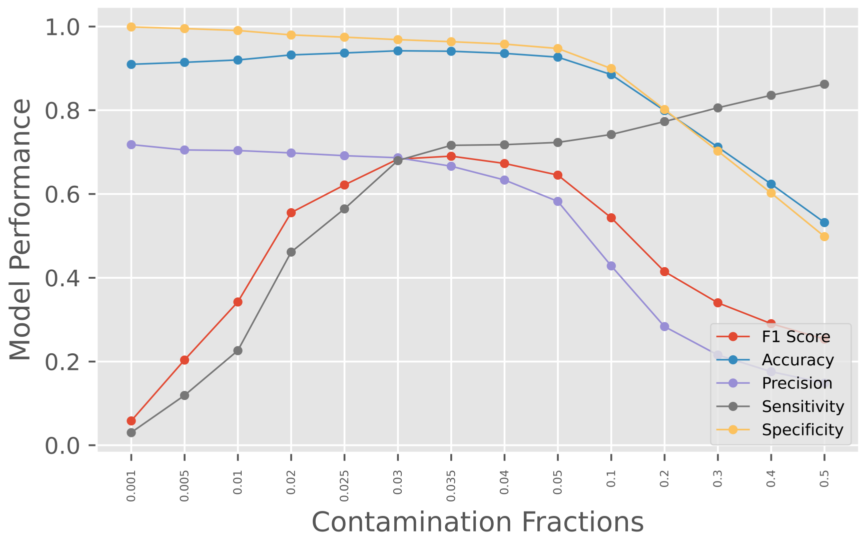


FIGURE A.36: Plot showing effect of contamination rate on model performance for 57-Bus-AM3 dataset.

APPENDIX B

A Damage Mitigation Model

B.1 INTRODUCTION

In this chapter, a model for damage mitigation when SFDI attacks take place without being detected and damage is either imminent or on the way is proposed.[6] The rationale for such a model is that despite the availability of sophisticated defense mechanisms, smart grid security breaches may happen causing serious damages. In the age of cheap and popular internet-enabled commodity products such as smart home devices, health monitors, vehicles, etc., attackers are finding increasingly more weakly-protected entry points to breach critical infrastructures such as smart grids. When fast changing attack signatures are used as a strategy to force the security apparatuses to relearn the attack properties and rethink countermeasures, critical time is lost and substantial damage can result. An attack strategy is considered that has the potentiality to breach smart grid security for which there is limited defense, and the detection of the attack takes substantially long time. It is proposed that in such scenarios, measures are deployed to isolate sub-networks to deny attacker's access to the most valuable assets.

[6]The work presented in this chapter has been published in the following paper:
M. Ashrafuzzaman, H. Jamil, Y. Chakhchoukh, and F.T. Sheldon, "A best-effort damage mitigation model for cyber-attacks on smart grids," 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), vol. 2, Pages 510-515, IEEE 2018. doi:10.1109/COMPSAC.2018.10285

B.2 THE GRIDWATCH MODEL

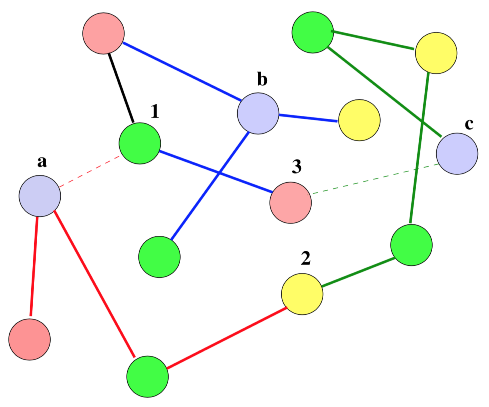
The assumption is that in the potentially compromised SG, the incidence response plan (IRP) no longer is reflective of a true defense or damage control plan for the entire SG although the IRP was developed meticulously with considerable expense. However, it may still be an effective plan for part of network and so it is worth implementing the plan for the pertinent sub-network. Because for those sub-networks, the IRP is still a valid response to any real failure. The proposed *GridWatch* damage mitigation model attempts to isolate sub-networks that can be self-sustaining according to the IRP in an attempt to shield it from cascading failures prematurely, or due to countermeasures deployed due to misleading state estimation (SE) by the main controller. Since the actual state of the SG is unreliable, a predicted time T is used that takes into account the time needed to detect attacks and estimated grid failure time, to isolate the SG into k independent sub-networks and temporarily relinquish control to the local controller so that the global state information does not affect the local operations. The choice of k will depend on the time-dependent importance or value of the sub-network, its role in the global SG, and its ability to sustain valuable services, within the predicted time T. In essence, the most valuable sub-networks is chosen to be saved first even if that means that a few low priority sub-networks have been compromised that could have been potentially saved.

In GridWatch, a smart grid is composed of a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}, \{G\} \rangle$ over a set of vertices $\mathcal{V}$. A vertex can represent any component of the smart grid, such as a power generator or power plant, load, microgrid, individual or a cluster of consumers, prosumer, sensor, transmitter, information and communication technology (ICT) tool, communication system, etc. The properties of a vertex will depend on the component it represents. Some of the common properties include capacity, type, criticality index, vulnerability index, etc.

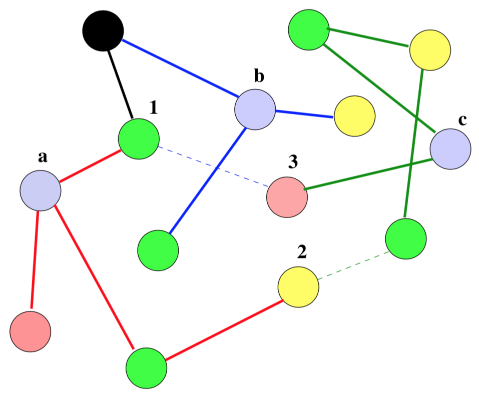
The set of edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ connects the components in a meaningful and functional way. There can be two kinds of edges: i) power-flow and ii) ICT connection. A power-flow edge will identify direction of power-flow between two components and the capacity of the power-line. Similarly, an ICT connection will represent if components are connected over network communication line. The FDI attacks can propagate only through the ICT edges, and cascading failures can only travel via the power-flow edges.

The set G is a set of graphs g of the form $\langle V, E \rangle$ over the sets $V \subseteq \mathcal{V}$ and $E \subseteq \mathcal{E}$ such that for any two graphs $g_1 = \langle V_1, E_1 \rangle$ and $g_2 = \langle V_2, E_2 \rangle$, $g_1, g_2 \in G$, the sets of vertices and edges are pairwise disjoint, i.e., $V_1 \cap V_2 = \emptyset$ and $E_1 \cap E_2 = \emptyset$. This implies that the graphs g_is are sub-networks that are not inter-connected. Furthermore, the sets $\mathcal{E}$ and E_is are pairwise disjoint as well, i.e., $\mathcal{E} \cap E_i = \emptyset$, for all i. That means, the edges in $\mathcal{E}$ help connect the sub-networks within the SG. Figure B.1(a) shows a GridWatch SG network $\mathcal{G}$ consisting of different types of resources, including controller nodes (a, b, and c) shown in blue. There are three sub-networks – red, blue and green (shown using edge colors). In this figure, red nodes are more valuable than the greens, and greens more than the yellow nodes. Figure B.1(a) shows three embedded graphs – blue, red and green. The thicker solid lines are active, and the thinner dashed lines are inactive but alternate connections. Figure B.1(b) shows re-routing of nodes a and 1, and disconnecting node 2 from the green graph.

The SG is also a tuple of the form $\langle \mathcal{G}, C, \Sigma, I, \Delta \rangle$, and so are each of the sub-networks $G_j \in \mathcal{G}$, where C is a controller, Σ is an SE, and I is an incident response plan (IRP) such that $\bigcup_j I_j \subseteq I$, i.e., the IRP for the SG is equal to or more that the union of all the



(a) Normally functioning SG.



(b) The black node is compromised.

FIGURE B.1: Model of GridWatch SG and possible countermeasure.

individual sub-network IRPs. The damage mitigation plan Δ is a function which aims to return a list of sub-networks S using a valuation function φ and a partition function π within a time budget T. The list S is computed such that for every member $s_i \in S$, for $i = 1, \ldots, |S|$, $\varphi(s_i) \geq \varphi(s_{i+1})$ at time t of the valuation. The valuation function φ is fundamentally an optimization function that maximizes the self-sustainability of s_i, the integrity of each sub-network $g \in G$ (keeping each g intact so that corresponding IRP I can remain effective), the importance of the network s_i in $\mathcal{G}$, and topology of $\mathcal{G}$.

However, the partition function π excludes all elements in $\mathcal{G}$ that are potentially compromised (based on sensor data and other network information), and creates smallest partitions or sub-networks that are self-sustainable. A sub-network is sustainable if they are spatially clustered, and includes all required SG components that are functional. The partition function potentially alters the topology of $\mathcal{G}$ since it considers all $v \in \mathcal{V}$ and all edges $e \in \mathcal{E} \cup \bigcup_i E_i$. The valuation function φ constructs sub-networks from the set of partitions returned by π, possibly fusing multiple partitions into one that share connecting edges or bottleneck partitions to maximize the value of the target sub-network. Finally, a sub-network is deemed self-sustaining if it includes generators and distribution systems sufficient to meet the energy demands of the consumers in the sub-network, has at least one controller, an SE, and sufficient IRPs. Also, the damage mitigation function Δ returns sub-network s_i sooner than s_j if $\varphi(s_i) \geq \varphi(s_j)$, for all values of i and j. To limit the possible damage to the sub-network, controller C disconnects s_i from the SG when returned by Δ, transfers control to C_i (the sub-network controller) with an instruction to disregard all control instructions until an authenticated instruction to relinquish control is received from the controller C, and serve as the sole independent controller of the sub-network s_i.

Figure B.1(b) shows the SG in Figure B.1(a) in which it was determined that the node marked black has been potentially compromised. To limit the potential damage, with the help of the functions π and φ, Δ reassigned node 1 under the control of node a, and since controller b cannot reach node 3 anymore, it needed to be reassigned. But it was determined by φ that reassigning it to controller c would result in a better self-sustaining sub-network system than assigning it to controller a.

B.3 CURRENT STATE OF GRIDWATCH

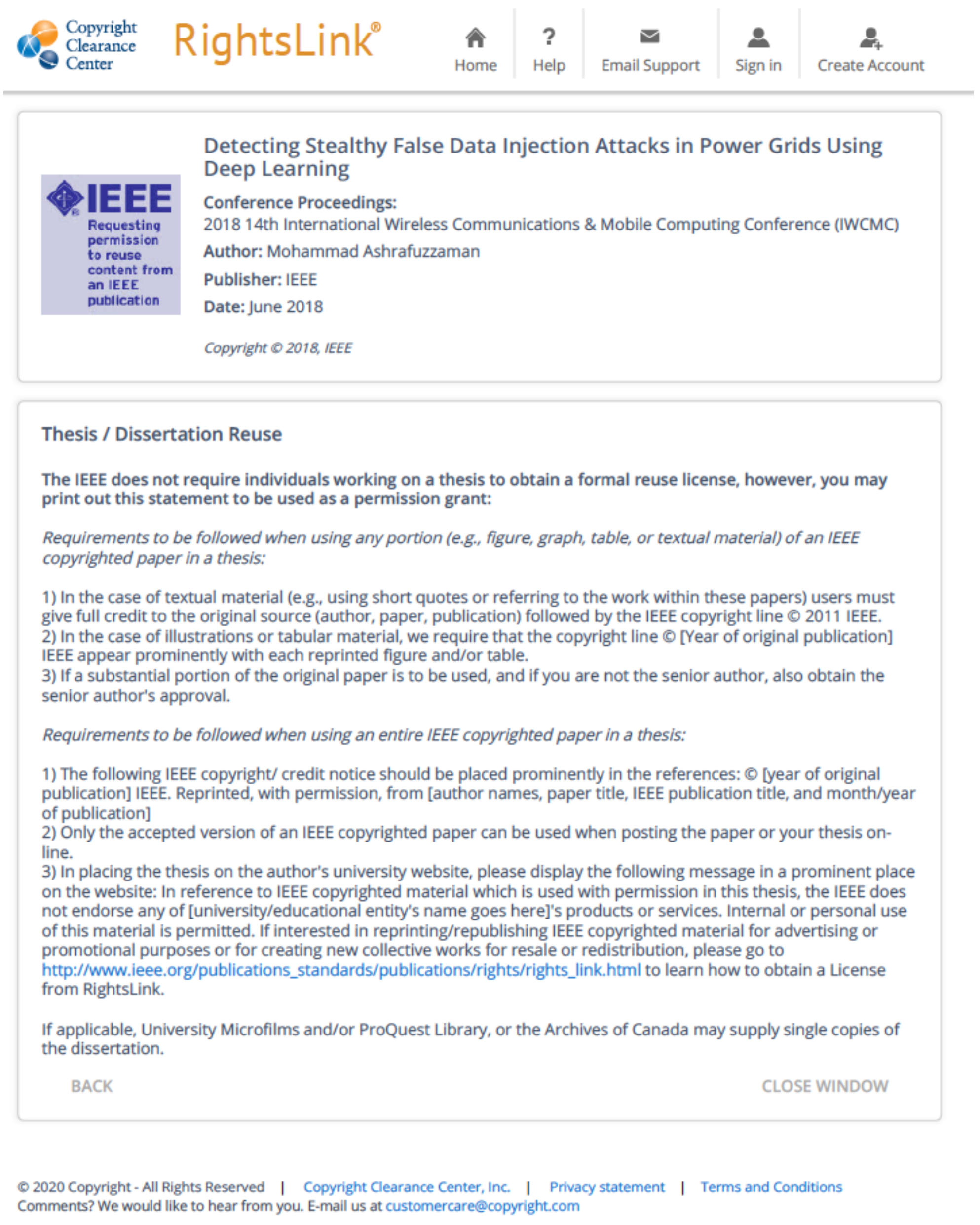
GridWatch is a conceptual model that aims to limit physical damage to SG, self-inflicted or otherwise, in real time. GridWatch recognizes the fact that there are many attack types, and not all attacks are recognizable, especially FDI attacks. It aims to buy time by isolating the grid into autonomous but smaller SGs so that proper detection and counter measures may be taken. It also admits some damage, if it must, to save the most valuable sub-networks from serious damages. In particular, GridWatch postulates that the spoofed data is a precursor of an attack, and the stealthy intent is to force the controller sustain self-inflicted wounds and make it damage the most-valuable sub-network.

B.4 SUMMARY

An outline of the conceptual model for a best-effort damage mitigation scheme is presented in this chapter. This conceptual model needs to be implemented and deployed on testbeds to evaluate its effectiveness.

APPENDIX C

PERMISSIONS TO REUSE PUBLISHED PAPERS


Copyright Clearance Center
RightsLink®
Home
Help
Email Support
Sign in
Create Account
IEEE Requesting permission to reuse content from an IEEE publication

Detecting Stealthy False Data Injection Attacks in Power Grids Using Deep Learning

Conference Proceedings:
2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)

Author: Mohammad Ashrafuzzaman

Publisher: IEEE

Date: June 2018

Copyright © 2018, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

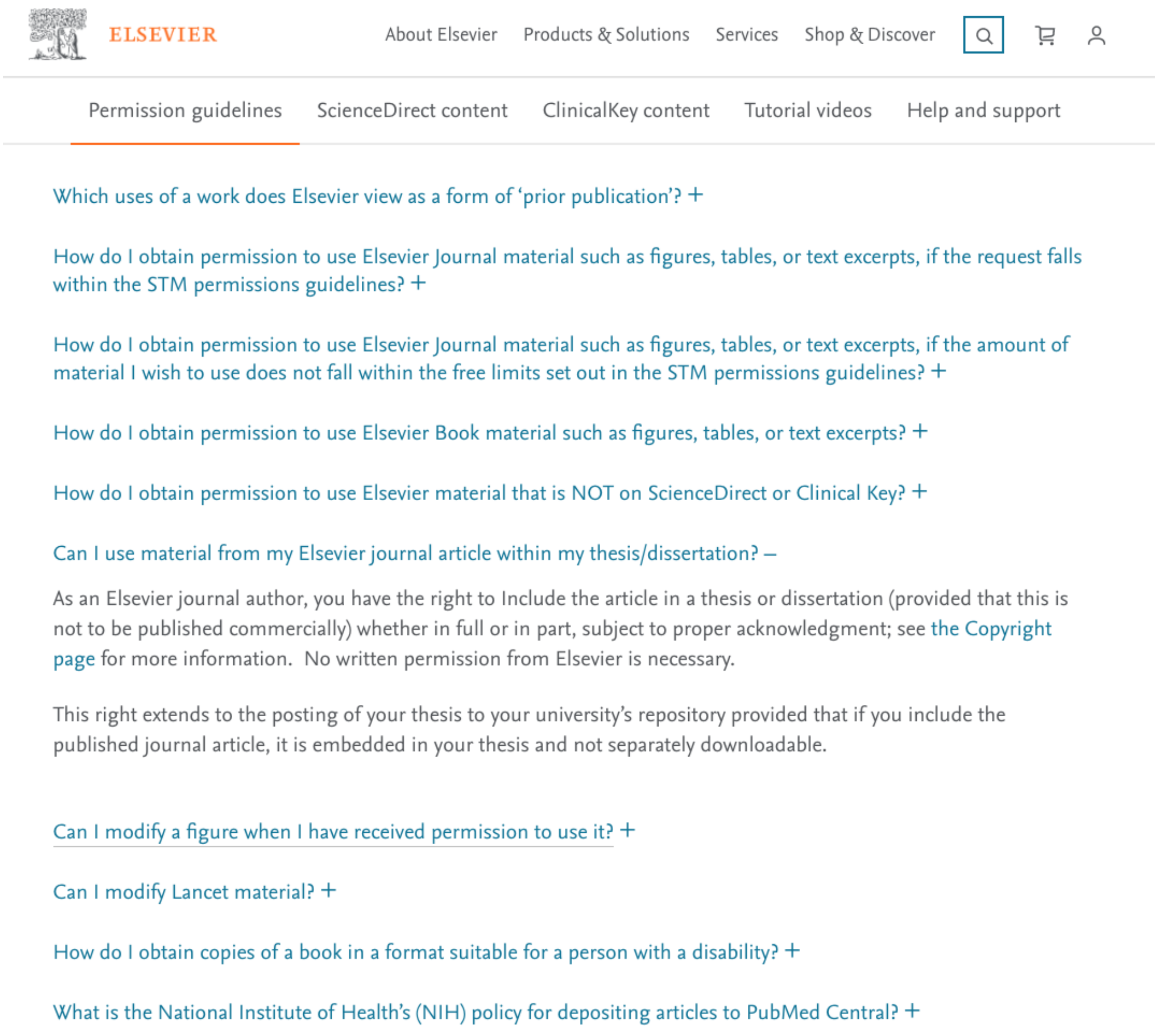
Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

© 2020 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | Privacy statement | Terms and Conditions
Comments? We would like to hear from you. E-mail us at customercare@copyright.com

FIGURE C.1: Permission from IEEE to reuse portion of my paper published in IEEE IWCMC 2018.



Which uses of a work does Elsevier view as a form of 'prior publication'? +

How do I obtain permission to use Elsevier Journal material such as figures, tables, or text excerpts, if the request falls within the STM permissions guidelines? +

How do I obtain permission to use Elsevier Journal material such as figures, tables, or text excerpts, if the amount of material I wish to use does not fall within the free limits set out in the STM permissions guidelines? +

How do I obtain permission to use Elsevier Book material such as figures, tables, or text excerpts? +

How do I obtain permission to use Elsevier material that is NOT on ScienceDirect or Clinical Key? +

Can I use material from my Elsevier journal article within my thesis/dissertation? –

As an Elsevier journal author, you have the right to Include the article in a thesis or dissertation (provided that this is not to be published commercially) whether in full or in part, subject to proper acknowledgment; see the Copyright page for more information. No written permission from Elsevier is necessary.

This right extends to the posting of your thesis to your university's repository provided that if you include the published journal article, it is embedded in your thesis and not separately downloadable.

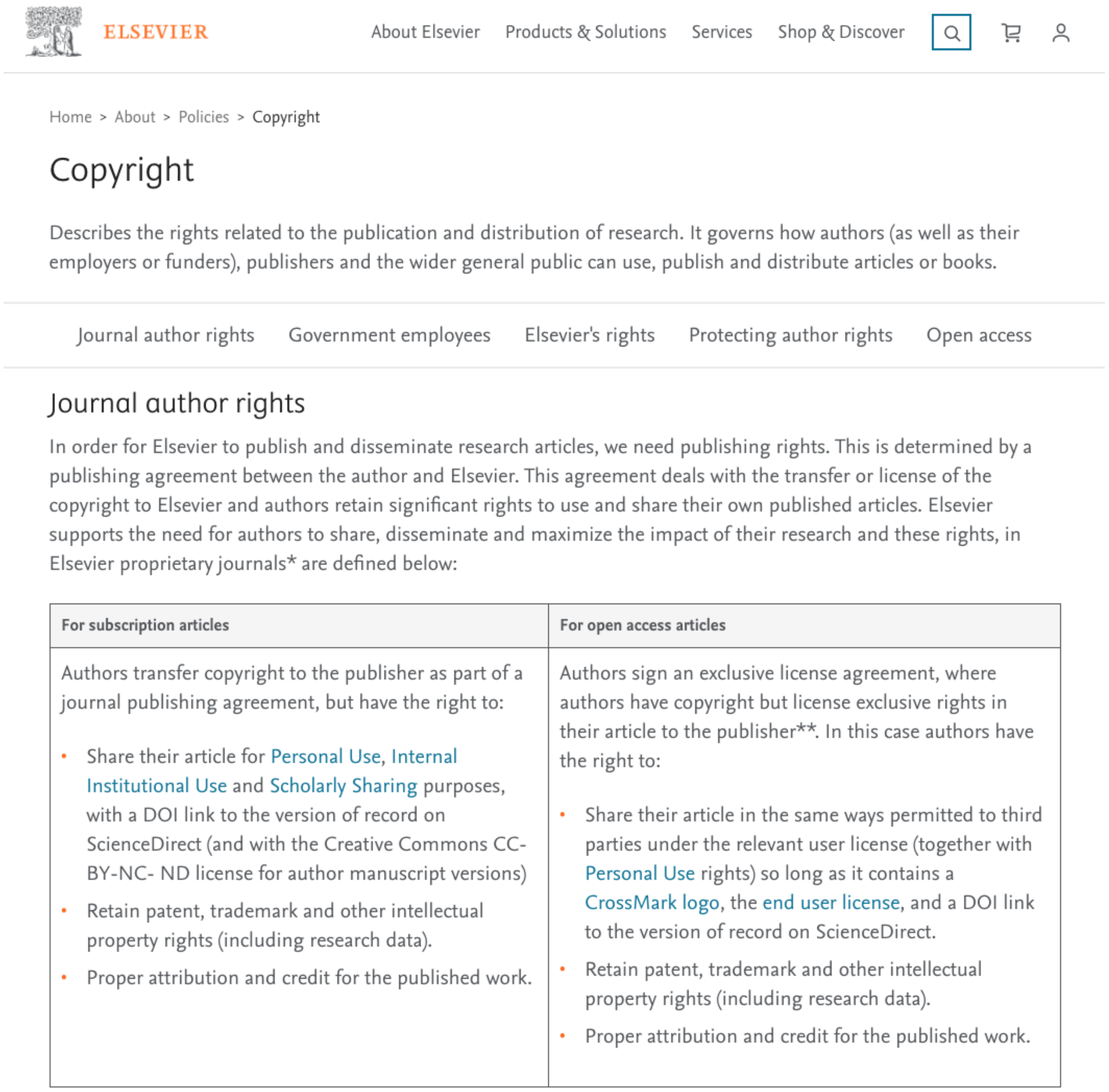
Can I modify a figure when I have received permission to use it? +

Can I modify Lancet material? +

How do I obtain copies of a book in a format suitable for a person with a disability? +

What is the National Institute of Health's (NIH) policy for depositing articles to PubMed Central? +

FIGURE C.2: Permission from Elsevier to reuse portion of my paper published in Computers & Security, 2020.



Copyright

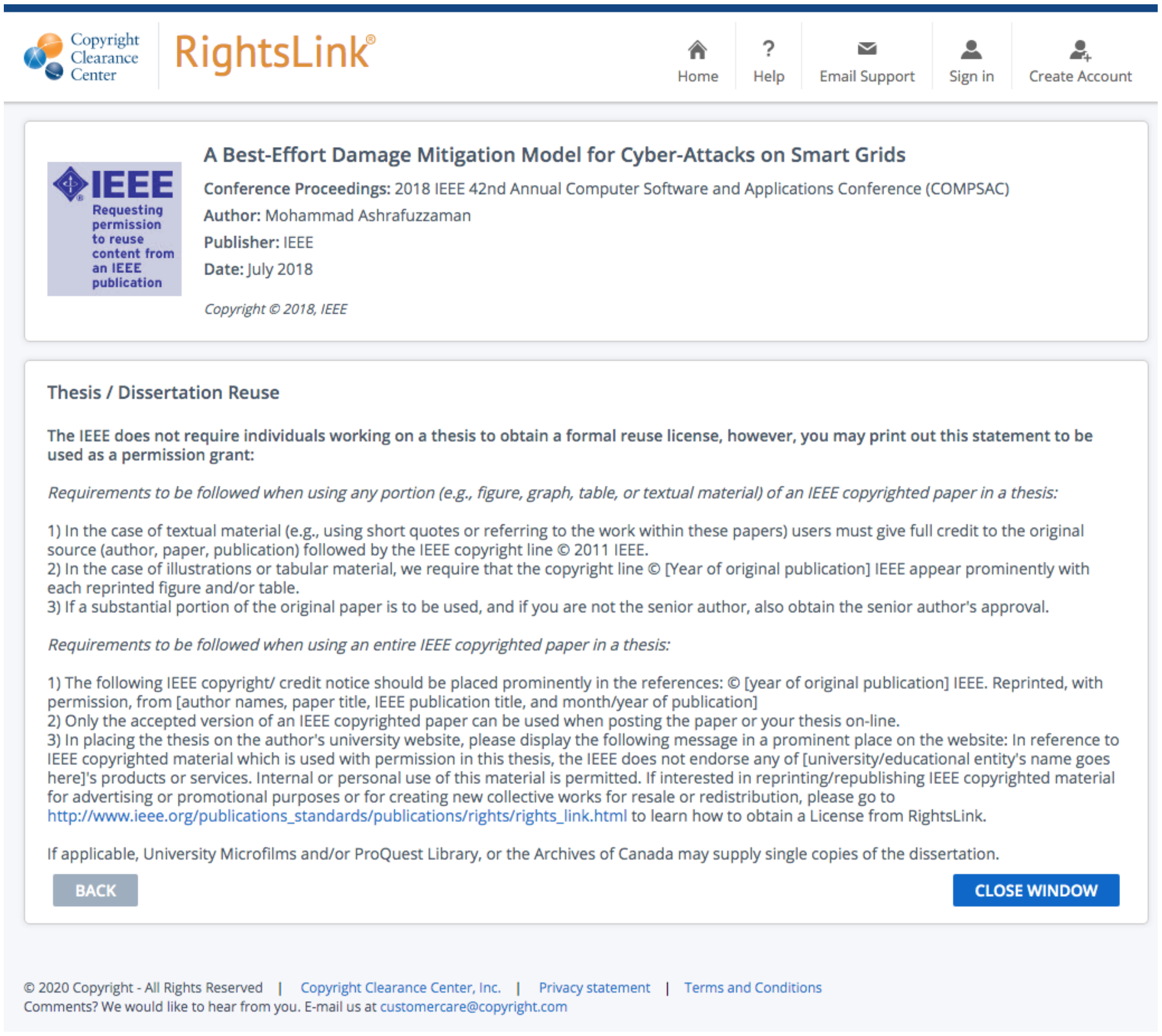
Describes the rights related to the publication and distribution of research. It governs how authors (as well as their employers or funders), publishers and the wider general public can use, publish and distribute articles or books.

Journal author rights

In order for Elsevier to publish and disseminate research articles, we need publishing rights. This is determined by a publishing agreement between the author and Elsevier. This agreement deals with the transfer or license of the copyright to Elsevier and authors retain significant rights to use and share their own published articles. Elsevier supports the need for authors to share, disseminate and maximize the impact of their research and these rights, in Elsevier proprietary journals* are defined below:

For subscription articles	For open access articles
Authors transfer copyright to the publisher as part of a journal publishing agreement, but have the right to: • Share their article for Personal Use, Internal Institutional Use and Scholarly Sharing purposes, with a DOI link to the version of record on ScienceDirect (and with the Creative Commons CC-BY-NC- ND license for author manuscript versions) • Retain patent, trademark and other intellectual property rights (including research data). • Proper attribution and credit for the published work.	Authors sign an exclusive license agreement, where authors have copyright but license exclusive rights in their article to the publisher**. In this case authors have the right to: • Share their article in the same ways permitted to third parties under the relevant user license (together with Personal Use rights) so long as it contains a CrossMark logo, the end user license, and a DOI link to the version of record on ScienceDirect. • Retain patent, trademark and other intellectual property rights (including research data). • Proper attribution and credit for the published work.

FIGURE C.3: Copyright guideline from Elsevier.


Copyright Clearance Center
RightsLink®
Home
Help
Email Support
Sign in
Create Account
IEEE
Requesting permission to reuse content from an IEEE publication

A Best-Effort Damage Mitigation Model for Cyber-Attacks on Smart Grids

Conference Proceedings: 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)
Author: Mohammad Ashrafuzzaman
Publisher: IEEE
Date: July 2018

Copyright © 2018, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

© 2020 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | Privacy statement | Terms and Conditions
Comments? We would like to hear from you. E-mail us at customercare@copyright.com

FIGURE C.4: Permission from IEEE to reuse portion of my paper published in IEEE COMSAC 2018.