Deep Latent Space Clustering for Detection of Stealthy False Data Injection Attacks Against AC State Estimation in Power Systems

Arnab Bhattacharjee, *Graduate Student Member, IEEE*, Arnab Kumar Mondal, Ashu Verma, *Senior Member, IEEE*, Sukumar Mishra, *Senior Member, IEEE*, and Tapan K. Saha, *Fellow, IEEE*

***Abstract*—This paper proposes a self-supervised latent space clustering algorithm, called the Deep Latent Space Clustering, for the detection of stealthy false data injection attacks (FDIAs) in smart grids against state estimation algorithms. The stealthy FDI model considered is based on the accurate AC state estimation and is able to bypass conventional bad data detection (BDD) algorithms with ease. The key element of the detection model is a stacked autoencoder network that first undergoes a carefully designed two-step finetuning process, following which a trainable clustering head is stacked on top of the finetuned encoder and the final network is further trained to achieve a clean clustering of the data into benign and compromised samples without labelled supervision. To test the efficacy and scalability of the detection model, it is tested on the standard IEEE 14 bus, 118 bus and 300 bus test systems. The self-supervised clustering model is compared to several supervised, semi-supervised and unsupervised algorithms proposed in the literature for detection of FDI on the aforementioned test cases and has been found to perform at par with the state of the art among them.**

***Index Terms*—Cyberattack detection, stealthy false data injection attacks, autoencoders, latent space clustering, AC state estimation.**

I. Introduction

THE INCORPORATION of advanced metering infrastructure, high-end computational resources and communication systems is set to turn the modern-day power system into a complicated but highly efficient and versatile cyber-physical system. Though there are many advantages of such a sophisticated system in achieving greater levels of control and observability over the electrical power network, they also make the grid susceptible to cyberattacks which pose a significant threat to its stability and reliability.

False Data Injection Attacks (FDIA) are one of the most malicious attacks carried out on cyber physical systems where an adversary injects perturbations in system measurements by leveraging communication channels or other vulnerable components of the network with an intention to degrade or disrupt the performance of the system. Power system operators generally employ residue-based Bad data detection (BDD) safeguards to detect the presence of such attacks in the system. However, intelligent adversaries can devise ways to bypass these conventional Bad data detection safeguards using limited system information to launch what are known as Stealthy False data injection attacks (SFDIA). In recent times, researchers have come up with numerous SFDIA models targeting different aspects of a power system. In this work, our focus is on SFDIA models targeting power system state estimation (PSSE) algorithms. Depending on whether the SFDIA is targeted towards DC state estimation or AC state estimation algorithms, they can be further classified into DC-SFDIA ([1], [2], [3], [4], [5], [6], [7]) and AC-SFDIA models ([8], [9], [10], [11], [12], [13]). Previous studies [8], [9] have shown that SFDIA vectors generated with the intention to bypass DC state estimation algorithms get easily detected by AC state estimation based BDD algorithms. Moreover, owing to the non-linear nature of AC state estimation, the amount of system information required to develop successful SFDIA vectors against AC state estimation is significantly greater than what is required for DC state estimation [9]. Hence, our primary focus is to develop advanced algorithms for the detection of stealthy FDI attacks built against AC state estimators. Such AC-SFDIA models can be built using complete ([8], [9], [10], [11]) or limited ([12], [13]) power system information, can be completely data-driven ([14], [15], [16]) or model-based ([17], [18]) and can be either used to target specific measurements [15], attack regions ([16], [17], [18]) or the entire power network [14]. In our work, we use two AC-SFDIA models that have been proposed in [17] and [19]. The attack model used by [17] requires solving an

Manuscript received 21 March 2022; revised 11 July 2022 and 11 September 2022; accepted 8 October 2022. Date of publication 25 October 2022; date of current version 21 April 2023. Paper no. TSG-00368-2022. *(Corresponding author: Arnab Bhattacharjee.)*

Arnab Bhattacharjee is with the Department of Electrical Engineering, The University of Queensland-IIT Delhi Academy of Research, Saint Lucia, QLD 4067, Australia (e-mail: arnab.bhattacharjee@uqidar.iitd.ac.in).

Arnab Kumar Mondal is with the Amar Nath and Shashi Khosla School of Information Technology, Indian Institute of Technology Delhi, New Delhi 110016, India (e-mail: anz188380@cse.iitd.ac.in).

Ashu Verma is with the Department of Energy Science and Engineering, Indian Institute of Technology Delhi, New Delhi 110016, India (e-mail: averma@ces.iitd.ac.in).

Sukumar Mishra is with the Electrical Engineering Department, Indian Institute of Technology Delhi, New Delhi 110016, India (e-mail: sukumar@ee.iitd.ac.in).

Tapan K. Saha is with the School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane, QLD 4072, Australia (e-mail: saha@itee.uq.edu.au).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSG.2022.3216625.

Digital Object Identifier 10.1109/TSG.2022.3216625

optimization problem to find a suitable SFDIA vector using complete information of the power system while targeting regions of varying radii whereas the attack model of [19] uses a heuristic data-driven algorithm to generate SFDIA vectors.

Numerous algorithms have been proposed in literature in the last decade for the detection of stealthy false data injections against AC state estimation in power systems [20]. These detection algorithms can be broadly classified into two groups – model based and model free. Model-based SFDIA detection algorithms are built on static and dynamic estimation techniques like Distributed Kalman Filters [21] and weighted least squares [22] or other more straightforward techniques like maximum likelihood estimation [23] or matrix separation [24] or multi-agent systems [25]. Extensive dependence on a precise system model and high sensitivity to its parameters, a high computational complexity and iterative nature are some of the major drawbacks that such model-based methods suffer from [20]. To overcome this dependence on system models, many works have proposed data-driven model-free algorithms based on concepts of statistical inference, machine learning and deep learning. Such model-free methods can be broadly classified into supervised and unsupervised categories.

A major segment of the existing literature on model free detection algorithms are supervised in nature, i.e., they require data samples that are labelled into attacked and benign classes for training purposes. Support vector machines (SVMs) are one of the most popular amongst supervised machine learning algorithms owing to their simplicity and versatility. SVMs identify the largest margin classifier and corresponding support vectors by minimizing a hinge loss to segregate data into different groups. Numerous research works have used SVMs for intrusion detection in power systems [26], [27], [28], [29], [30]. The task of choosing kernel functions is, however, a major bottleneck for SVMs. Other popular supervised machine learning algorithms that have been used for SFDIA detection include Decision Trees [30], K Nearest Neighbors [29], Random Forests [29], Gradient Boosting [31] and Naive Bayes' Classifiers [32]. The authors in [33] propose a supervised statistical method to detect attacks against distributed state estimation in cyber-physical systems where an Epanechnikov kernel based kernel density estimator is employed to obtain the the distributions of attacked and benign state values following which statistical parameters like moments, skewness and kurtosis are collected as feature vectors to feed into a Median Absolute deviation-based outlier detection algorithm. Although being powerful pattern recognition paradigms, these statistical machine learning algorithms often suffer from one or more of the following limitations – high sensitivity to hyperparameters and noisy outliers, distributional assumptions over data and limited function approximation capacity due to lower number of trainable parameters.

Overcoming these limitations, specialized multi-layered neural networks gave rise to a class of learning algorithms called deep learning algorithms that have gained widespread popularity in various fields of engineering and sciences as powerful function approximators. They encompass Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Autoencoders(AEs), Generative Adversarial Networks(GANs), Graph Neural Networks (GNNs) and many such powerful architectures. CNNs apply spatial or temporal convolutions to input data in the form of images to extract localized representative features. RNNs and its advanced variants like Long Short Term Memory (LSTM) and Gated Recurrent Units (GRU), are capable of extracting long term dependencies in sequential data using sophisticated hidden states that are updated over a temporal dimension. These two popular classes of algorithms – CNNs ([29], [34]) and RNNs ([35], [36]) - have been widely used for SFDIA detection. The authors of [37] use a combined and explainable convolutional and LSTM architecture for cyberthreat detection in Industrial IoT. An autoencoder based supervised SFDIA detection algorithm is proposed in [38]. The autoencoder is first trained to reconstruct its input using the benign samples. The authors claim that when an attacked vector is given as an input to the trained AE, the reconstruction error given by the AE will be larger as compared to benign samples. The authors also propose a supervised Denoising autoencoder to extract correct measurements from the attacked ones. A combined autoencoder and GAN based semi-supervised detection algorithm is proposed for power distribution networks in [39]. Using a small amount of labelled data the authors train an autoencoder in an adversarial fashion for binary classification of benign and attacked samples. In [40], the authors employ a conditional Deep Belief Network to identify the temporal patterns in data following which a binary classification head classifies the data as attacked or benign. In order to obtain a greater number of attacked data samples, the authors train generative models on existing labelled attacked samples to artificially generate more datapoints. The method is tested on a DC-SFDIA model.

GNNs are special types of deep neural networks that use generic forms of the convolution operator to process data supported on non-euclidean structures like graphs and trees. Considering bus measurements as data supported on the nodes of the power network, authors in [17] employ an Auto-Regressive Moving Average GNN or ARMA – GNN based classification algorithm for joint detection and localization of SFDIA attacks.

A much discussed limitation of these deep learning algorithms is that they need a lot of time to train. But it is important to understand that in most cases, the training of these algorithms is carried out offline with historical data, following which they are used in an online fashion for real-time detection. Once trained, deep learning algorithms act very fast, in the order of milliseconds, as they only need to perform basic feedforward matrix operations which can be done very efficiently by modern day computational devices. So a large training time doesn't offer any hindrance to the real-time performance of deep learning algorithms in general.

However, considering the long training times as a potential limitation, researchers have proposed alternatives like extreme machine learning algorithms that randomly assign weights and biases. In [18], the authors propose a joint SFDIA detection and state recovery algorithm. FDIA infected data samples are identified using a trained ensemble of supervised extreme

machine learning modules, following which attack localization is carried out by comparative analysis of measured PMU values and forecasted state values. Once the attacks is localized, a quasi-Newton method and Armijo line search is employed for state recovery in the identified locations. This method requires a lot of data measurements from both SCADA and PMUs, thus limiting its widespread applicability.

Although being great advancements to the application of SFDIA detection, all the aforementioned data-driven methods are supervised in nature, i.e., they require labelled data for training. However, labelling of datapoints is either difficult or highly expensive to carry out and requires substantial human time, effort and intervention. Moreover, supervised algorithms tend to easily overfit the training data, leading to suboptimal performance in unseen domains. These are some of the most significant limitations of supervised data-driven algorithms. Unsupervised algorithms, on the other hand, can be trained to extract rich distinguishing features from raw unlabelled data. These features can then be used downstream for clustering the data into different groups. It is to be noted here that nowhere during the training of the models, either during feature extraction or the downstream clustering task, does a completely unsupervised algorithm need any form of labelled information. Due to these constraints, unsupervised learning is in general an ill-posed task and is a difficult feat to achieve. Therefore, very few unsupervised machine learning and deep learning algorithms have been presented for the detection of stealthy false data injections in existing literature. Amongst them, in the works that use statistical machine learning ([41], [42], [43]), the dimensionality of the datapoints is first reduced via supervised feature selection or unsupervised dimensionality reduction algorithms. The data with reduced dimensions is then segregated into attacked and benign samples using unsupervised clustering or anomaly detection algorithms. The authors of [41] use a supervised random forest classifier for selecting the most relevant features from the data samples. Following this, they apply an Elliptic Envelope based anomaly detection algorithm for identifying attacked data samples. This method suffers from two major limitations. Firstly, it performs poorly with non-gaussian distributed data and secondly, the Elliptic Envelope algorithm needs to exactly know the level of contamination in the dataset to perform optimally. A similar approach is followed in [42] where Principal Component Analysis (PCA) is used for dimensionality reduction, following which an Isolation Forest based anomaly detection algorithm is used for attack detection. On similar lines, the Isolation Forest based anomaly detection algorithm functions with an assumption that the content of anomaly in the dataset is significantly low as compared to the amount of benign data. Hence their performance degrades if the anomaly content in the data is increased. The authors in [43] use a two level dimensionality reduction strategy where statistical features like moments, skewness, mean and kurtosis are extracted from the attacked and benign data samples in a supervised fashion followed by further dimensionality reduction using PCA. The processed data samples are then segregated using the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm into attacked and benign classes.

The use of completely unsupervised deep learning algorithms has been very limited for the application of SFDIA detection. A recent work that is most similar to our proposed method was carried out by the authors of [19] who proposed a Deep Autoencoding Gaussian Mixture Model (DAGMM) based anomaly detection algorithm. The measurement data samples are transformed to a different feature space using a deep autoencoder, following which a Gaussian Mixture model is fit to the data samples and an energy function value is computed for each sample. The data samples with a higher energy as compared to a calculated threshold value are labelled anomalous. Although the process till the computation of the energy of the data samples is unsupervised, label information is required for calculating the energy threshold that is used for anomaly detection. Hence, this method cannot be considered as completely unsupervised.

Our work aims to address the research gaps existing in current literature as discussed above. In the following list, we enumerate our key contributions:

1) We propose a completely unsupervised Deep Latent Space Clustering (DLSC) algorithm for accurate and fast detection of stealthy FDIA in smart grids without the need for any information about network parameters or measurement distributions.
2) *Completely unsupervised:* Unlike other unsupervised baselines like [19], [41], [43], our method doesn't require label information at any stage of the training or real-time testing process. This bypasses the dependence on the highly expensive and error prone task of manual data labelling.
3) *Robust, versatile and generalizable:* The performance of our method is insensitive to hyper-parameters and the same architecture can be used effectively for a wide range of applications - including for different SFDIA attack models and across datasets with varying anomaly percentages, unlike ([41], [42]). Its performance is also independent of the training data distribution, unlike [41], and can be easily generalized to unseen domains.
4) *Rapid detection:* The proposed attack model is fast. The training time required maybe marginally higher than the existing models but that is carried out offline before online deployment. Once deployed online for detection of SFDIA against the PSSE, the run-time of the algorithm is of the order of a few milliseconds.
5) *Scalable and accurate:* The proposed detection algorithm is scalable and achieves state of the art performance on the IEEE 14, 118 and 300 bus test systems on different supervised and unsupervised metrics. It is a significant result given the fact that the method doesn't require any external labelled information.

The rest of the paper is organised in the following manner. Section II provides a brief introduction to the building blocks of this paper. It covers AC state estimation, stealthy FDIA model and the conventional BDD algorithms. The proposed method is detailed in Section III. Section IV consists of a description of the test systems, the false data generation process and other experimental details. Section V

contains the results and inferences. The paper is concluded in Section VI.

II. Background

Here we consider an electrical power network with N buses $\in \mathcal{N}$ and M lines $\in \mathcal{M}$. The voltage magnitude and angle values at the i^{th} bus where $i \in \mathcal{N}$ are given by V_i and θ_i respectively. The active and reactive power injections at each bus is denoted by P_i and Q_i respectively $\forall\ i \in \mathcal{N}$. The active and reactive power flows at each line $(i,j)\ \forall (i,j) \in \mathcal{M}$ and $i,j \in \mathcal{N}$ are given by P_{ij} and Q_{ij} respectively. The network topology and line parameters are represented by the network admittance matrix $\boldsymbol{Y} \in \mathbb{C}^{N\times N}$. The real and imaginary parts of Y_{ij} are denoted by G_{ij} and B_{ij} respectively. Vectors are written in smallcase bold letters, matrices are represented using bold uppercase letters and the rest are scalars. The $(i,j)^{th}$ element of a matrix $\boldsymbol{A}$ is given by A_{ij}.

A. AC State Estimation

State estimation is one of the most important tasks carried out in a power system control centre. The estimated states play a key role in deciding the values of control or operational parameters in downstream tasks in the power system. In the steady state, the power system can be modelled using non-linear power flow equations mapping the power injections and line flows to voltage magnitudes and angles as shown below:

$$\begin{aligned} P_i &= V_i \sum_{j=1}^{N} V_j\big(G_{ij}cos\big(\theta_i-\theta_j\big)+B_{ij}sin\big(\theta_i-\theta_j\big)\big) \\ Q_i &= -V_i \sum_{j=1}^{N} V_j B_{ij}cos\big(\theta_i-\theta_j\big)-\big(G_{ij}sin\big(\theta_i-\theta_j\big)\big) \end{aligned} \tag{1}$$

$$\begin{aligned} P_{ij} &= V_i^2 G_{ij} - V_i V_j\big(G_{ij}cos\big(\theta_i-\theta_j\big)+B_{ij}sin\big(\theta_i-\theta_j\big)\big) \\ Q_{ij} &= -V_i^2 B_{ij} - V_i V_j\big(G_{ij}sin\big(\theta_i-\theta_j\big)-B_{ij}cos\big(\theta_i-\theta_j\big)\big) \end{aligned} \tag{2}$$

The voltage magnitude and angles at each bus constitute the state $\boldsymbol{x}$ of the electrical network. The power injections at each bus and the sending and receiving end line flows constitute the measurement vector $\boldsymbol{z}$. In a power system control center, a noisy version of these measurements is collected from the meters to estimate the state values of the network. The noise in the measurements originates from measurement devices, channel imperfections, etc. Following this, an AC Power system state estimation(AC-PSSE) is carried out by solving the following weighted least squares estimation problem:

$$\hat{\boldsymbol{x}} = \min_{\boldsymbol{x}}\Big[(\boldsymbol{z}-h(\boldsymbol{x}))^T \boldsymbol{R}^{-1}(\boldsymbol{z}-h(\boldsymbol{x}))\Big] \tag{3}$$

where $\boldsymbol{R}$ is the error covariance matrix of the measurement vector, $h(.)$ represents the non-linear power flow equations 1 and 2, and $\hat{\boldsymbol{x}}$ represents the estimated state vector. Since $h(.)$ is a non-convex function, this optimization problem can not be solved in closed form. Hence, we use an iterative weighted least squares algorithm to obtain the estimated states.

B. Stealthy False Data Injection Attacks

When an attacker carries out a FDIA, the goal that they want to achieve is to add a malicious vector to the noisy measurements obtained from the meters in such a way that the AC-PSSE converges to a different state value than expected.

$$\hat{\boldsymbol{z}} = \boldsymbol{z} + \boldsymbol{a} = h\big(\hat{\boldsymbol{x}} + \boldsymbol{c}\big)$$

The vector $\boldsymbol{a}$ is called the attack vector and $\hat{\boldsymbol{z}}$ represents the compromised measurement vector following an attack. In stealthy FDIA attacks the malicious data content is added carefully so as to bypass the LNRT based BDD mechanism with a high probability. In an AC-PSSE model, the sufficient condition that the additive attack vector $\boldsymbol{a}$ needs to satisfy to carry out a stealthy attack is given by [9]:

$$\boldsymbol{a} = h\big(\hat{\boldsymbol{x}} + \boldsymbol{c}\big) - h\big(\hat{\boldsymbol{x}}\big)$$

The primary attack model we adopt in our work is a faster batch implementation of a generic false data injection method employed in [17], [44]. In this model, we assume that the attacker has complete information of the power network and thus can launch attacks over regions of varying radii. Hence, we initially choose an attack region randomly from the entire network. Since attacking at generator nodes and zero injection buses can lead to easier detection, we assume that the attacker targets only load buses. The attack region consists of a load bus randomly chosen with probability p_{tg}, and its k-hop neighboring load buses. Once all the nodes to be attacked are identified, they are included in the set $\mathcal{T}_{bus}$. Afterwards all the lines connecting the nodes in $\mathcal{T}_{bus}$ are included in $\mathcal{T}_{line}$. The attacker can only manipulate the measurements from within the attack region defined by $\mathcal{T}_{bus}$ and $\mathcal{T}_{line}$. The attacker then solves the following optimization problem to generate compromised state vectors:

$$\begin{aligned} \widehat{\boldsymbol{x}_a} = \underset{\widehat{\boldsymbol{x}_a}}{\operatorname{argmin}}\ & \beta_z \|h\big(\hat{\boldsymbol{x}}\big)_i - h(\widehat{\boldsymbol{x}_a})_i\| - \beta_x \|\hat{\boldsymbol{x}}_i - \widehat{\boldsymbol{x}_a}_i\| \\ & \forall i \in (\mathcal{T}_{bus}, \mathcal{T}_{line}) \end{aligned} \tag{4}$$

s.t.

$$\hat{\boldsymbol{x}}_l = \widehat{\boldsymbol{x}_a}_l \forall l \notin (\mathcal{T}_{bus}, \mathcal{T}_{line}) \tag{5}$$

$$h\big(\hat{\boldsymbol{x}}\big)_l = h(\widehat{\boldsymbol{x}_a})_l \forall l \notin (\mathcal{T}_{bus}, \mathcal{T}_{line}) \tag{6}$$

$$\tau_x^{min} \leq \|\widehat{x_a}_i\| \leq \tau_x^{max} \tag{7}$$

Here, $\widehat{\boldsymbol{x}_a}$ is the compromised estimated state vector, $\hat{\boldsymbol{x}}$ is the original estimated state vector, τ_x^{min} and τ_x^{max} are the minimum and maximum limits of the state vector.

The first term in (4) minimizes the gap between the original measurement vector and the compromised measurements by minimizing the L1 loss between them, thus increasing stealthiness. And the second term in (4) maximizes the L1 loss between the estimated normal and compromised state vectors, thus increasing attack strength. It is solved using a gradient based iterative approach like stochastic gradient descent. The constraints ensure that values are changed only in the attack region and that the variables stay within their limits. After the optimization is complete, the attacker checks the error value between the compromised and original measurements and allows them into the system as false data only if the

measurement vector residual error is below a threshold τ_{loss}. The attacker repeats this procedure every instant they want to launch an attack into the system. This attack model is referred to as the optimization-based attack model in the subsequent sections.

C. Conventional Bad Data Detection Algorithms

The following set of equations constitute a conventional BDD algorithm:

$$\begin{aligned} \boldsymbol{r} &= |\boldsymbol{z} - h(\hat{\boldsymbol{x}})| \\ \boldsymbol{K} &= \boldsymbol{H}^T \boldsymbol{R}^{-1} \boldsymbol{H} \\ \boldsymbol{B} &= \boldsymbol{I} - \boldsymbol{H}\left(\boldsymbol{K}\boldsymbol{H}^T \boldsymbol{R}^{-1}\right) \\ r_i^{norm} &= \frac{|z_i - h(\hat{x})_i|}{R_{ii} B_{ii}} \end{aligned} \tag{8}$$

The current state vector is estimated using a PSSE module following which the measurement error residues are calculated using equation (8). $\boldsymbol{H}$ represents the power flow Jacobian matrix and $\boldsymbol{R}$ is the diagonal error covariance matrix of the measurement vector. They are used to obtain the normalized residual vector $\boldsymbol{r}^{norm}$. The largest normalized residue of the measurement vector $\boldsymbol{z}$, given by $LNR(\boldsymbol{z}) = max(\boldsymbol{r}^{norm})$ is then compared against a predefined threshold τ_{bdd}. If the largest residue is greater than the threshold, it implies that the measurement vector has been corrupted. Therefore, a hypothesis testing is carried out where the null and alternative hypotheses are given by $\mathcal{H}_0$: Attack has not taken place and $\mathcal{H}_1$: System has been attacked.

$$\begin{aligned} &\mathcal{H}_1 \\ &max(\boldsymbol{r}^{norm}) \gtrless \tau_{bdd} \\ &\mathcal{H}_0 \end{aligned} \tag{9}$$

This test is also known as the Largest Normalized Residual Test (LNRT). A stealthy FDIA attack vector is a type of malicious attack vector which bypasses the LNRT test, i.e., for which $max(\boldsymbol{r}^{norm}) < \tau_{bdd}$.

III. Proposed Detection Strategy

Clustering is a method of data analysis that groups data objects together based on similarity. The curse of dimensionality often arises when clustering high-dimensional data. In high dimensions, data points can be very close to each other but still appear to be isolated and lead to clusters that are too small or too large and do not accurately reflect the similarities among the data points. To alleviate this problem, we propose to use a Deep Latent Space Clustering algorithm that first transforms the input data, $\boldsymbol{o}^{in}$ from the original feature space, $\mathcal{O}$ with a non-linear mapping, E_ϕ to extract relevant features $\boldsymbol{s} \in \mathcal{S}$. These transformed features, also known as the latent space features obtained through a sophisticated training algorithm, form natural clusters in $\mathcal{S}$. Autoencoders are a natural choice for learning this non-linear transformation as they offer an unsupervised framework for representation learning. Fig. 1 presents a simplified block diagram of the proposed method.

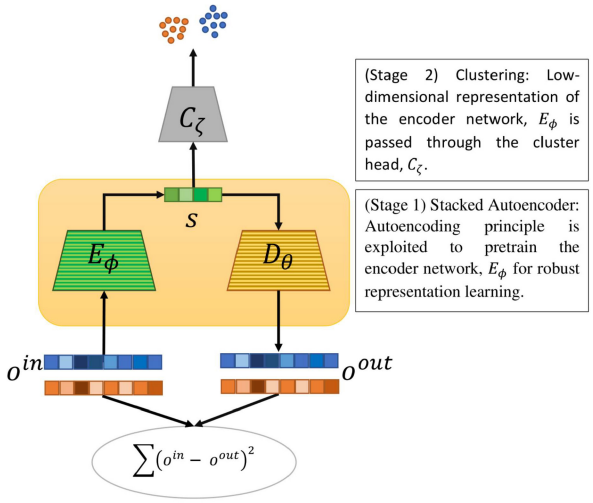


Fig. 1. Block Diagram of the proposed method.

A. Stacked AutoEncoder

Representation learning is a subfield of machine learning concerned with learning efficient representations of data. A popular approach for unsupervised representation learning is to use stacked autoencoders. A stacked autoencoder is a neural network consisting of multiple perceptron layers as shown in Fig. 2. The output of each hidden layer is connected to the input of the successive hidden layer. This strategy allows the model to learn progressively more complex data representations. Fig. 2 depicts a detailed step by step breakdown of the training procedure involved in the proposed method. The SAE network is first initialised layer by layer in step 1 of the proposed algorithm. Each encoder layer and its corresponding decoder layer constitutes a denoising autoencoder that is trained with an aim of reconstructing the output of the previous layer following random corruption. A denoising autoencoder is a two layer neural network that is trained using a reconstruction loss between its input, $\boldsymbol{o}^{in}$ and output $\boldsymbol{o}^{out}$ and is defined as:

$$\begin{aligned} \tilde{\boldsymbol{o}}^{in} &= Dropout(\boldsymbol{o}^{in}) \\ \boldsymbol{g} &= \sigma_1\left(W_1 * \tilde{\boldsymbol{o}}^{in} + b_1\right) \\ \tilde{\boldsymbol{g}} &= Dropout(\boldsymbol{g}) \\ \boldsymbol{o}^{out} &= \sigma_2(W_2 * \tilde{\boldsymbol{g}} + b_2), \end{aligned}$$

where $Dropout(.)$ refers to the stochastic dropout function that randomly masks some of its inputs, σ_1 and σ_2 are non-linear activation functions and W_l and b_l are the weights and biases corresponding to layer l. The weights of the autoencoder are updated using backpropagation on the L2 loss given by $\sum_i (\boldsymbol{o}_{in}^{(i)} - \boldsymbol{o}_{out}^{(i)})^2$. The layer wise training procedure is greedy in nature as the outermost encoder and decoder layers are trained first before passing the encoder output for training the subsequent encoder and decoder layers. After a greedy layer-wise training, the encoder and reversed decoder layers are concatenated to build a deep autoencoder, which is then finetuned to minimize an L2 reconstruction loss between the input and

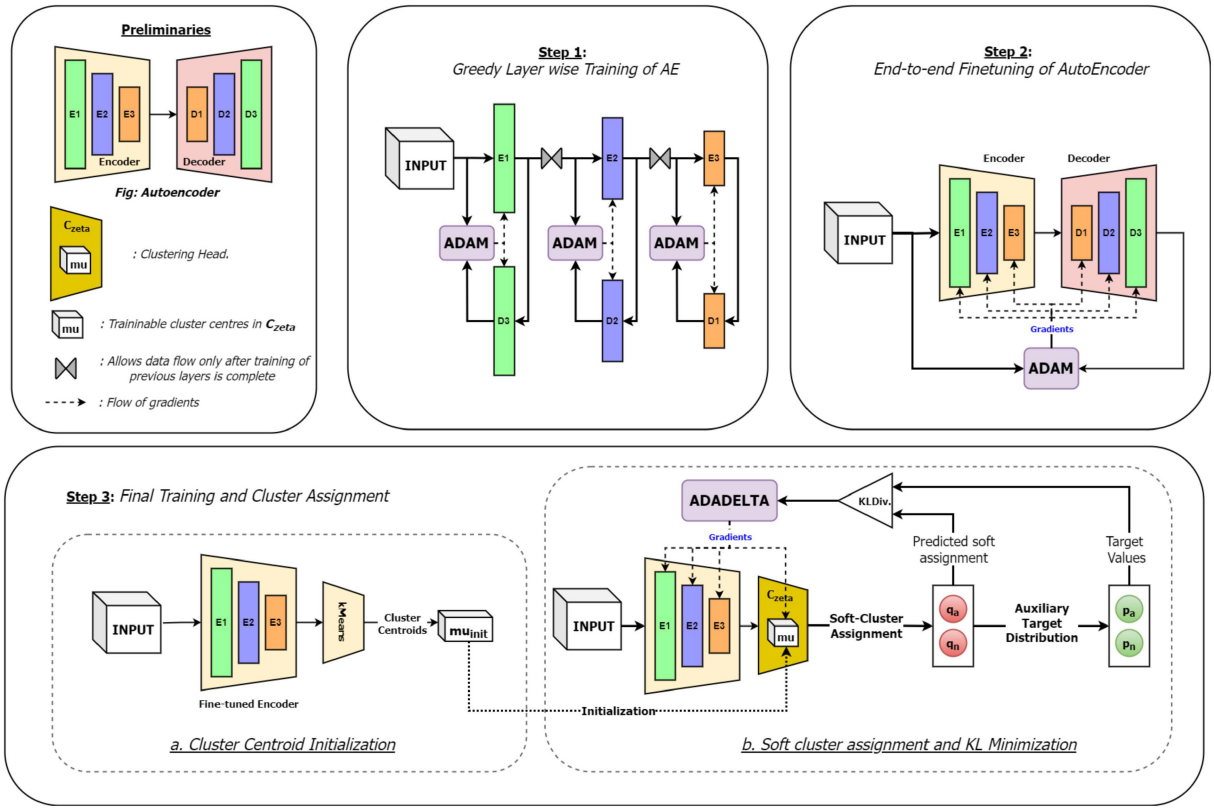


Fig. 2. Detailed operational breakdown of the proposed method.

the autoencoder output. This comprises step 2 of the algorithm as depicted in Fig. 2. The final outcome is a multi-layer deep autoencoder with a bottleneck coding layer in the middle. Thereafter, the decoder layers are removed, leaving only the encoder layers to serve as an initial mapping between the original and latent feature spaces.

B. Self Training for Improving Cluster Assignment

This subsection corresponds to the step 3 as shown in Fig. 2. The fine tuned encoder obtained after step 2, E_ϕ, serves as the initial estimator of the non-linear mapping from the original feature space to the latent space, i.e., $E_\phi : \mathcal{O} \rightarrow \mathcal{S}$. Next, the k-means clustering algorithm is applied on the low dimensional latent space features, $\boldsymbol{s} = E_\phi(\boldsymbol{o}^{in})$, to compute the initial cluster centers $\{\boldsymbol{\mu}_j\}_{j=1}^{k}$ where k is the number of clusters which is 2 in our case representing the attacked and benign classes.

These initial cluster centers are used to initialize the trainable cluster centroids of the clustering head C_ζ. Once the clustering head is initialized, a two-step unsupervised algorithm is employed to improve the cluster assignment as shown in step 3b in Fig. 2. First, a soft assignment between the latent space features and the cluster centres is computed. In the second step, we use an auxiliary target distribution to learn from existing high confidence assignments to update the encoder parameters and refine the cluster centroids. These two steps are alternatively repeated until the convergence condition is met.

1) Probabilistic Cluster Assignment: Based on the similarity between the latent space feature, $\boldsymbol{s}^{(i)}$ corresponding to the i^{th} data sample, and the cluster center, $\boldsymbol{\mu}_j$, the data sample i is assigned a probability, q_{ij} indicating the extent of its membership to the j^{th} cluster where $i \in [1, 2, \ldots, |\mathcal{D}|]$ and $j \in [1, .., k]$. Here, $\mathcal{D}$ represents the dataset used to train the model and $|\mathcal{D}|$ represents the number of samples in the dataset and k represents the total number of clusters. A Student's t-distribution based similarity kernel is used for q_{ij} computation as follows:

$$q_{ij} = \frac{\left(1 + \frac{||\boldsymbol{s}^{(i)} - \boldsymbol{\mu}_j||^2}{\gamma}\right)^{-\frac{\gamma+1}{2}}}{\sum_{j'}\left(1 + \frac{||\boldsymbol{s}^{(i)} - \boldsymbol{\mu}_{j'}||^2}{\gamma}\right)^{-\frac{\gamma+1}{2}}} \tag{10}$$

where, γ represents the degrees of freedom of the student's t-distribution. The value of γ is fixed at 1 for all of our experiments.

2) Kullback–Leibler Divergence Minimization: In this step depicted by step 3b in Fig. 2, we further refine the trained weights of the encoder E_ϕ and the trainable cluster centroids corresponding to the clustering head C_ζ by minimizing the Kullback Liebler Divergence between the soft cluster assignments, following a distribution $\mathbb{Q}$, and an auxiliary target distribution $\mathbb{P}$. The KL Divergence quantifies the dissimilarities between two probability distributions and is given by:

$$\mathcal{L} = KL(\mathbb{P}||\mathbb{Q}) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{11}$$

Careful consideration needs to be put to choose an appropriate target distribution. A simple approach would be to choose a one hot encoding type of delta distribution where for every $\boldsymbol{s}^{(i)}$,

p_{ij^*} is one iff $j^* = min\{j : \|\boldsymbol{s}^{(i)} - \boldsymbol{\mu}_j\| \forall j \in [1, \dots k]\}$, otherwise zero. However such hard assignments may not be suitable in this case as the cluster assignments themselves represent soft probability values. Hence, in this work, we define the auxiliary target distribution as below:

$$p_{ij} = \frac{\frac{q_{ij}^2}{y_j}}{\sum_l \left(\frac{q_{il}^2}{y_l}\right)} \tag{12}$$

where $y_l = \sum_i q_{il}$ where q_{il} are the soft assignments corresponding to the i-th datapoint. The trainable parameters, Φ, of the encoder E_ϕ and the cluster centres of C_ζ are then updated based on the KL Divergence as follows:

$$\begin{aligned}
\Phi &= \Phi - \eta_\phi \frac{\partial \mathcal{L}}{\partial \boldsymbol{s}^{(i)}} \frac{\partial \boldsymbol{s}^{(i)}}{\partial \Phi} \\
\boldsymbol{\mu}_j &= \boldsymbol{\mu}_j - \eta_\mu \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_j} \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{s}^{(i)}} &= \left(\frac{1+\gamma}{\gamma}\right) \sum_{j=1}^{k} \left(1 + \frac{||\boldsymbol{s}^{(i)} - \boldsymbol{\mu}_j||^2}{\gamma}\right)^{-1} \\
&\quad \times \left(p_{ij} - q_{ij}\right)\left(\boldsymbol{s}^{(i)} - \mu_j\right) \\
\frac{\partial \boldsymbol{s}^{(i)}}{\partial \phi} &= E'_\phi\left(\mathbf{x}_{in}^{(i)}\right) \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_j} &= \left(\frac{1+\gamma}{\gamma}\right) \sum_{i=1}^{|\mathcal{D}|} \left(1 + \frac{||\boldsymbol{s}^{(i)} - \boldsymbol{\mu}_j||^2}{\gamma}\right)^{-1} \\
&\quad \times \left(p_{ij} - q_{ij}\right)\left(\boldsymbol{s}^{(i)} - \boldsymbol{\mu}_j\right)
\end{aligned} \tag{13}$$

Convergence is assumed to have been attained if the difference between the cluster assignments is less than a level of tolerance in consecutive epochs. In Fig. 2, the nodes $q(p)_a$ and $q(p)_n$ represent the probabilities corresponding to the student's t - distribution(target distribution) for the attacked and benign clusters respectively. Since $k = 2$, the subscripts a and n correspond to the two clusters.

IV. Case Study

A. Test Systems

Due to unavailability of rich attack datasets on power systems, we generate attack data using three standard IEEE test systems - 14 bus, 118 bus and 300 bus systems. The 300 bus test case has been included to demonstrate the scalability of the method. The simulations were carried out in Python and Pandapower. To impart realistic characteristics to the simulations, real life load profiles for a year with a sampling interval of one hour is collected. Fig. 3 shows the load profile data sampled hourly for a week. The load profile is first normalized to zero mean and unity standard deviation and is represented by $\mathcal{U}$. For every time instant t, we replace the scaling factors of the generator and load buses in the test systems by randomly sampling from a normal distribution with mean $1 + v * \mathcal{U}_t$, where $v \in (0, 1]$ and standard deviation σ_s. The scaling range is limited between 0.7 and 1.3. Then AC power flow is run to obtain the voltage magnitudes, angles, power injections and line flows. A 1% noise is then added to the power injections and line flow values to obtain the noisy measurement vector. In the next step, an AC PSSE is conducted to find the estimated states. The measurement and estimated state vector samples generated after this step are represented by $\mathbf{Z} \in \mathbb{R}^{T \times N_z}$ and $\hat{\mathbf{X}} \in \mathbb{R}^{T \times N_x}$. T represents the length of the time horizon, N_z and N_x represents dimensionality of each measurement and state vector respectively. In our case, N_x is equal to $2 * N$ and N_z is equal to $(2 * N + 4 * M)$ where N and M are the total number of buses and lines in the network respectively. The dataset generated after this step is constituted of benign samples $\mathbf{Z}$ and $\hat{\mathbf{X}}$. In the following section a portion of the benign data samples would be used to create attack vectors.

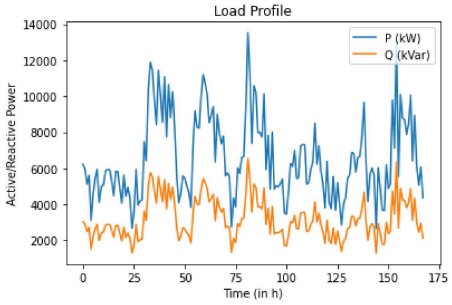


Fig. 3. Hourly load profile over a week.

B. Stealthy FDIA Data Generation

The original optimization based SFDIA generation model can be found in detail in [17], [44]. We adopt a faster batch implementation of the algorithm for our work. After the benign state and measurement vectors are obtained following the steps described in Section IV-A, we randomly select some time instances, with a probability p_{att}, from within the time horizon as the instances of plausible FDI. We collect these instances into the set $\mathcal{T}_{attack}$. For every instance $t \in \mathcal{T}_{attack}$, we randomly select an attack region by following the steps described in Section II-B. The assault mechanism is then invoked to generate the compromised state vector $\widehat{\boldsymbol{x}_a}^t$ by solving the optimization problem given by equations (4)–(7). The assault mechanism starts by defining a tuple consisting of two trainable variables $0.9 \leq V_t \leq 1.1$ and $-\pi \leq \theta_t \leq \pi$ that represent the attacked states. These trainable vectors are initialized with a random sample drawn from a normal distribution with mean equal to the benign voltage magnitude and angle values corresponding to the time step t respectively and standard deviation equal to 0 for buses outside the attack region and a standard deviation of σ_n for the buses inside the attacked region. It is important to ensure that these trainable attack vectors are initialized in proximity to the original state vectors, failing to do which might lead to problems in convergence. The trainable vectors V_t and θ_t are then updated over a fixed number of epochs using a gradient based optimizer like Adam or SGD with an aim to minimize the objective function given in equation (4) while satisfying the constraints. After the training procedure is complete, the value of the first term of the objective function (4), denoted by $\mathcal{L}_z$ and given

TABLE I
PARAMETERS FOR FDIA ATTACK GENERATION

Parameters	Notations	Values
Time Horizon	T	[1 to 8760]
Std. dev. of noise added to initialize **V** and θ	σ_n	0.005
Loss coefficients	β_x, β_z	1, 1
Optimizer learning rate	η	0.001
Max. no. of epochs to train **V** and θ	MaxEpochs	2000
Loss threshold for acceptance as attack post training	τ_{loss}	0.1
Minimum attack radius	R_{min}	{14:2, 118:4, 300:6}
Maximum attack radius	R_{max}	{14:4, 118:10, 300:12}
Scaling factor coeff.	v	0.1
Scaling factor standard deviation	σ_s	0.01

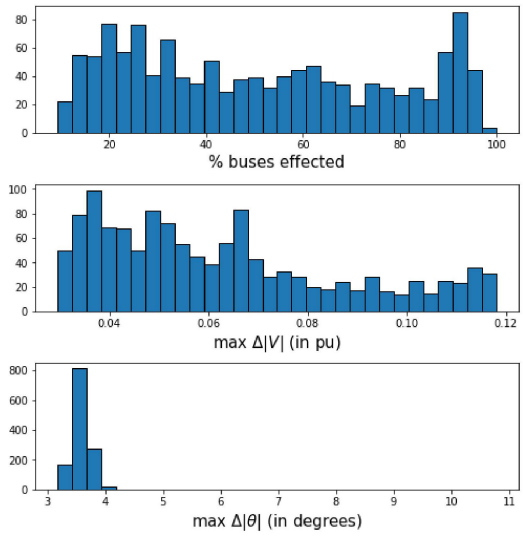
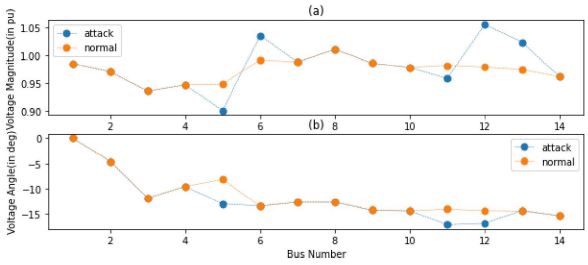


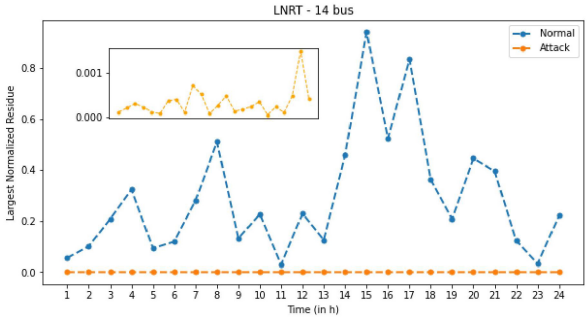
Fig. 4. Distribution of attack scale incurred by the optimization based attack model in the form of the number of attacked buses and maximum state deviations caused.

by $\mathcal{L}_z = \|h(\hat{\boldsymbol{x}}^t) - h(\hat{\boldsymbol{x}}_a^t)\|$ is analysed and the attack vector generation is considered successful if $\mathcal{L}_z \leq \tau_{loss}$. The attacker computes the corresponding corrupted measurement values, $\boldsymbol{z}_a^t$, using $\boldsymbol{z}_a^t = h(\hat{\boldsymbol{x}}_a^t)$ and injects the false data in the system only when an attack vector is successfully generated. This process is repeated for all the time instances in $\mathcal{T}_{attack}$. The values of the different parameters required for attack generation are given in Table I.

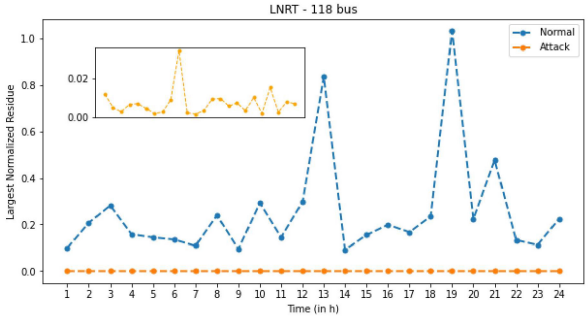
The distribution of the number of buses affected and the maximum state deviation caused by the optimization based attack model is given in Fig. 4 for the IEEE 118 bus case. The attacks are spread to varied spatial extents in the network as the number of buses and their corresponding states effected by the attack model ranges from 10% to 100%. State deviations caused by the attacks are also significantly high, ranging from 0.01 to 0.12 p.u. for voltage magnitudes and 1 to 11 degrees for voltage angles. Thus the optimization based attack model used in this work is capable of launching attacks of varying strengths and over a varied number of components of the power network, including full scale network attacks. Fig. 5(a) depicts the difference in state values before and after a stealthy FDIA attack on the IEEE 14 bus network.



(a) Changes in state values for each bus in the IEEE 14 bus system before and after a stealthy FDIA attack.(a): Voltage Magnitude(in pu). (b): Voltage Angle (in degrees)



(b) Largest Normalized Residue values for the IEEE 14 bus case under attacked and normal condition for a period of 24 hours.



(c) Largest Normalized Residue values for the IEEE 118 bus case under attacked and normal condition for a period of 24 hours.

Fig. 5. Characteristics of the attack vectors generated using the optimization based attack model.

Fig. 5(b) and Fig. 5(c) depict the results of the LNR test for detection of Bad Data using the conventional BDD algorithm on the IEEE 14 bus and 118 bus test systems. As is clear from the results, the stealthy FDI attack generated using the optimization based attack model can very easily bypass a LNR-based BDD algorithm.

C. Deep Latent Space Clustering Based Detection

For the purpose of detection using the proposed Deep Latent Space Clustering algorithm, we make use of only the active and reactive power injections at each bus as the input features. All other measurements that have been mentioned apriori, like line flows, are not used for training the detection model. The SAE model we adopt has three layers each in its encoder and decoder networks. The dimensionality of the hidden layers is 512 and the output layer is 10. The greedy layer wise training is carried out for three pairs of corresponding encoder and decoder layers and each training session is run for 25000 steps

TABLE II
COMPARISON WITH BASELINES

Methods	Type	IEEE 14 bus				IEEE 118 bus				IEEE 300 bus			
		Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
SVM [26]	Supervised	0.8765	0.9983	0.7549	0.8597	0.7887	0.8364	0.6721	0.7453	0.7904	0.7865	0.6484	0.7108
RNN [47]	Supervised	0.8738	0.9607	0.7838	0.8633	0.8656	0.9556	0.7473	0.8387	0.8087	0.7917	0.6449	0.7108
CNN [29]	Supervised	0.8767	0.9636	0.793	0.87	0.9114	0.936	0.8633	0.8982	0.9493	0.9444	0.9211	0.9326
Cheb-GNN [44]	Supervised	0.9064	0.9729	0.8397	0.9014	0.958	0.978	0.9061	0.9407	0.9819	0.9886	0.9651	0.9767
ARMA-GNN [17]	Supervised	0.9972	0.9958	0.9982	0.997	0.993	0.984	0.9736	0.9788	0.9974	0.9905	0.9897	0.9901
AAE [39]	Semi-Supervised	0.9588	0.9379	0.96	0.9488	0.9653	0.9471	0.9685	0.9577	0.9631	0.9526	0.9688	0.9606
k-Means	Unsupervised	0.7491	0.8221	0.7499	0.7843	0.4517	0.4223	0.4506	0.4359	0.5187	0.482	0.4811	0.4815
kPCA('poly', 10) + kMeans	Unsupervised	0.7538	0.8195	0.7602	0.7887	0.4627	0.4239	0.4663	0.444	0.5346	0.52	0.5379	0.5287
kPCA('rbf') + kMeans	Unsupervised	0.76	0.8277	0.7712	0.7984	0.4533	0.4219	0.4574	0.4389	0.52	0.52	0.52	0.52
TPN [45]	Unsupervised	0.9453	0.9593	0.935	0.947	0.9528	0.9662	0.9313	0.9484	0.9583	0.9592	0.9459	0.9525
Random Forest Feature Selection + Elliptic Envelope [41]	Unsupervised	0.6588	0.8241	0.5404	0.6527	0.632	0.7747	0.5298	0.6222	0.6791	0.8346	0.5429	0.6578
PCA + Isolation Forest [42]	Unsupervised	0.7608	0.7593	0.7608	0.76	0.6899	0.7025	0.6331	0.6659	0.6561	0.6313	0.6562	0.6435
PCA + DBSCAN [43]	Unsupervised	0.6485	0.8037	0.5269	0.6365	0.952	0.9502	0.9534	0.9518	0.9468	0.9419	0.9498	0.9458
DAGMM [19]	Unsupervised	0.9175	0.8047	0.9362	0.8655	0.9584	0.909	0.9491	0.9286	0.9582	0.8713	0.9728	0.9192
DLSC (Proposed Method)	Unsupervised	**0.9999**	**0.9999**	**0.9999**	**0.9999**	**0.9986**	**0.9992**	**0.9987**	**0.9989**	**0.9989**	**0.9988**	**0.999**	**0.9989**

where each step consisted of training the network with a data batch size of 256. The next step, i.e., the end-to-end finetuning of the complete stacked autoencoder, is run for 50000 steps with a mini batch size of 256. The optimization algorithm used for the first two steps corresponding to Figure 2 is Adam and for the 3rd step, Adadelta optimizer is used. The 3rd step is run for 100000 steps with a minibatch size of 256 per step. We used the Tensorflow library to implement the proposed DLSC. A key point to note here is that we assume that the power system operator has access to unlabelled system data consisting of attack scenarios. Also, since instances of attacks are rare, it has also been assumed that amongst the two clusters, the one with greater number of samples represents the benign case and the cluster with smaller number of data points represent the attacked case.

D. Performance Metrics

$$Accuracy = \frac{tp + tn}{N_p + N_n}$$
$$Precision = \frac{tp}{tp + fp}$$
$$Recall = \frac{tp}{tp + fn}$$
$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

The performance of our model is tested on several supervised metrics. We use the data labels only during the evaluation phase for calculating the performance metrics. For comparison purpose, we use the Accuracy, Precision, Recall, and F_1 metrics. Let tp, fp, tn fn, N_p and N_n represent the count of true positives, false positives, true negatives, false negatives, actual total count of positive samples and actual total count of negative samples following segregation of data points respectively. Then, the supervised metrics are defined as follows:

All these metrics together quantify the quality of segregation of attacked and benign data points. Precision, Recall and F_1 metrics are essential to understand the performance of a segregation model when the distribution of the number of samples belonging to the different categories is skewed.

E. Baselines

Two major classes of experiments are carried out. In the first set of experiments, we compare our method against numerous state-of-the-art model-free SFDIA detection algorithms on the optimization-based attack model. Since our proposed method is unsupervised in nature, we primarily choose to compare the performance of our method against other state of the art unsupervised baselines like Elliptic Envelope [41], Isolation Forest [42], DAGMM [19], DBSCAN [42] and Transferrable Prototypical Networks (TPNs) [45]. As a standard unsupervised baseline, we also present the detection performance of the k-Means algorithm and its non-linear transformed versions. For the non-linear versions of k-Means, we transform the data points from their original feature space to a non-linear feature space using kernel-PCA (kPCA) with polynomial and radial basis kernels, prior to clustering. To identify the best performing hyperparameters for the kernel-PCA, we tested on different hyperparameters and found almost similar results and decided to go with the following for their slightly better performances:

1) number of components as 10 for the 14 bus system, 20 for the 118 bus system and 50 for the 300 bus system for both the kernels, and
2) number of degrees as 10 for the polynomial kernel.

We also present comparisons of our method against popular supervised algorithms like SVMs ([26], [27], [28], [29], [30], RNNs [35], [36]), CNNs ([29], [34]), and two forms of GNNs – Chebyshev GNN and ARMA-GNN, proposed in [44] and [17] respectively. Other than that, a semi-supervised detection algorithm using Adversarial Autoencoders [17] is also considered as a baseline. In the first set of results, we compare all these detection algorithms with respect to their accuracy, precision, recall and F1 values on the task of detecting the presence of SFDIA attacks modelled using the optimization-based attack model on three IEEE test systems – 14 bus, 118 bus and 300 bus systems. The results corresponding to these tests have been presented in Table II.

TABLE III
COMPARISON OF SFDIA DETECTION PERFORMANCE OF UNSUPERVISED BASELINES ON DIFFERENT ATTACK MODELS AND UNDER DIFFERENT IMBALANCES

	Optimization Based Attack Model [17]								Data driven Attack Model [19]							
	1:2				1:8				1:2				1:8			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
DAGMM [19]	0.9584	0.909	0.9491	0.9286	0.9228	0.8254	0.951	0.8837	0.9765	0.9872	0.9677	0.9773	0.9668	0.9977	0.965	0.981
Random Forest Feature Selection + Elliptic Envelope [41]	0.632	0.7747	0.5298	0.6222	0.9056	0.8911	0.8625	0.8766	0.6443	0.7738	0.5347	0.6324	0.8878	0.8965	0.8687	0.8824
PCA + Isolation forest [42]	0.6899	0.7025	0.6331	0.6659	0.9145	0.9466	0.9145	0.9303	0.6163	0.5894	0.6163	0.6025	0.8265	0.8774	0.8265	0.8512
PCA + DBSCAN [42]	0.952	0.9502	0.9534	0.9518	0.949	0.924	0.9536	0.9385	0.9845	0.9831	0.9855	0.9843	0.9853	0.9735	0.9872	0.9803
DLSC	**0.9986**	**0.9992**	**0.9987**	**0.9989**	**0.9977**	**0.9982**	**0.9972**	**0.9977**	**0.9968**	**0.9973**	**0.9985**	**0.9979**	**0.9995**	**1**	**0.9995**	**0.9979**

For the second set of experiments, in addition to the optimization based attack model, a completely data driven heuristic based SFDIA model as described in [19] is also tested upon. The performance of the unsupervised baselines ([19], [41], [42], [43]) are compared with our proposed method under the two different attack models and with varying anomaly percentages, represented by the imbalance ratio (IR). The imbalance ratio is defined as the ratio of total compromised samples to the number of benign samples in the dataset. We perform these experiments with two IR values: 1:2 and 1:8. These tests are carried out on the IEEE 118 bus test system as it has been considered widely across literature as a standard test system for SFDIA detection algorithms. The results corresponding to the second set of experiments is presented in Table III.

t-SNE Visualization: In order to get a better understanding of the performance of the Proposed Method, we use visualization tools to create 2D visual maps of the original feature space, $\mathcal{Z}$ of measurements z, and the transformed feature space, $\mathcal{S}$, learned by DLSC. The visual maps are created by transforming the data using the t-Students Distributed Stochastic Neighbour Embedding or t-SNE algorithm. The t-SNE algorithm projects datapoints from a high dimensional space into a 2D space suitable for visualization. Further details about t-SNE can be found in [46]. For a qualitative analysis, we obtain the t-SNE projections of the data to visualize the distribution of the corrupted and benign samples:

1) in the ground truth data in the original feature space, $\mathcal{Z}$,
2) identified using the standard kMeans algorithm in the original feature space, $\mathcal{Z}$,
3) in the transformed data space $\mathcal{S}$ learned by the encoder of the proposed DLSC algorithm and after applying the learned clustering head, C_ζ.

We carry out the t-SNE visualization experiments with two different imbalance ratios - 1:2 and 1:8 and present the results in Figures 6 and 7 respectively. We also compute the Normalized Mutual Information(NMI) and Adjusted Rand Index(ARI) scores for the kMeans algorithm and the proposed method in the t-SNE visualization plots:

1) *Normalized Mutual Information (NMI):* NMI is an information theory principled measure that captures the reduction in entropy of output labels given cluster labels. NMI is formally defined as:

$$\text{NMI} = \frac{I(Y_{gt}; Y_{out})}{\sqrt{H(Y_{gt})H(Y_{out})}} \tag{14}$$

where $I(Y_{gt}; Y_{out})$ signifies mutual information between ground truth labels (Y_{gt}) and the output labels (Y_{out}). $H(Y_{gt})$ and $H(Y_{out})$ denote the entropy of ground truth and output labels respectively.

2) *Adjusted Rand Index (ARI) score:* ARI is a metric to quantify similarity between two clustering results. ARI is defined as:

$$\text{ARI} = \frac{\text{RI} - \mathit{expected}(\text{RI})}{\mathit{max}(\text{RI}) - \mathit{expected}(\text{RI})} \tag{15}$$

where *RI* denotes rand index. Rand index calculates similarity between two clustering results by comparing the ground truth and output labels of all possible pair of elements from the dataset. NMI and ARI are clustering performance metrics and the closer they are to 1, the better is the clustering performances.

F. Time Complexity

One of the major challenges of deep learning based algorithms is the long training times that they require. Classical machine learning algorithms take much less time to train. The time required to train our proposed method using a free Google Colab single GPU system is 50 minutes as compared to around a minute for machine learning based algorithms. The training time is large compared to statistical methods because during training, the model error is fed back to update the large number of trainable weights using backpropagation. Hence the training is carried out offline prior to on-field deployment. However, once the model is trained, only feedforward matrix operations are required to be done and hence the test time computational complexity drops significantly. The saved model can be directly used for detection using the trained encoder and clustering head. Thus, in real-time testing phase, our model can process a batch of thousand datapoints within 10 milliseconds, which is similar to other machine learning algorithms. This is because after training, the only computations that need to be carried out are matrix operations like addition and multiplication in a feed-forward fashion, which can be done very efficiently and rapidly using any modern day computational devices. The authors of [19], who proposed a deep learning based unsupervised SFDIA detection algorithm, also arrive at a similar conclusion.

V. RESULTS

Here we infer from the results observed by performing the experiments laid out in Section IV-E. The performance of

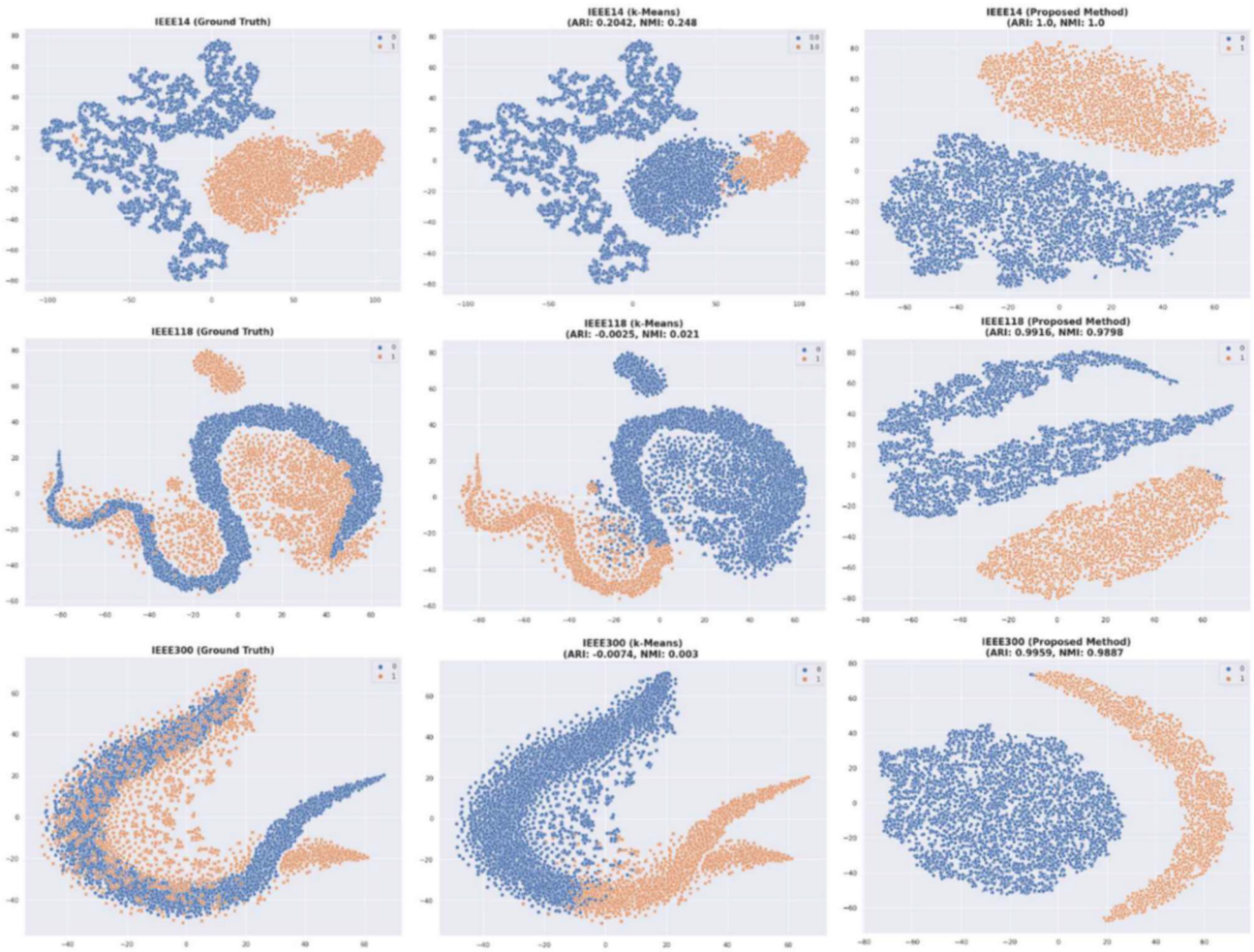


Fig. 6. Visualization of t-SNE projections of the ground truth data (1st column of each row), clusters obtained using k-Means baseline algorithm(2nd column of each row) and clusters obtained using our proposed method (3rd column of each row) when the imbalance ratio is approximately equal to 1:2. The three rows correspond to the IEEE 14 bus, 118 bus and 300 bus test systems respectively. The data is inseparable in the original feature space but forms clean clusters in the latent variable space.

the various detection baselines and the proposed method on the optimization-based attack model are presented in Table II. The IR value for these first set of experiments is kept fixed at 1:2. The proposed method performs significantly better than the other unsupervised baselines in terms of Accuracy, Precision, Recall and F1 scores. The only other baseline whose performance comes close to that of the proposed method is that of the ARMA-GNN based detection algorithm [17], which is a powerful supervised framework for graph data analysis. The standard kMeans clustering algorithm fails to correctly segregate the benign samples from the attacked ones in the original feature space. Similar results are observed even when the kMeans clustering is applied to polynomial and radial-basis function transformed data, implying that the attacked and benign samples do not form linearly separable clusters in the original feature space or in the non-linearly transformed space mapped using the polynomial or radial-basis kernels. This is further corroborated by the t-SNE visualizations given in Figures 6 and 7 where it can observed that in the original feature space, the attacked and benign samples are highly intertwined, resulting in the failure of the kMeans clustering algorithms. However, the proposed method is able to clearly segregate the attacked samples from the benign ones in the latent space, $\mathcal{S}$, learned by the encoder. This signifies the importance of transforming the datapoints to the learned latent space, $\mathcal{S}$, for attaining separability. Table III captures the comparison of the performance of the proposed method and the other unsupervised detection baselines over data generated using the optimization-based attack model and the heuristic data driven attack model with different imbalance ratios. The limitations of the various unsupervised baselines can be observed through these results. The Random Forest Feature selection and Elliptic Envelope based detector [41] performs significantly better when the IR is 1:8 as compared to when its 1:2 under both the attack models. One of the most significant hyperparameters of the Elliptic Envelope based anomaly detection algorithm is the contamination rate, which signifies the percentage of anomaly in the dataset. The contamination rate in all these experiments has been fixed at 0.03 as was deduced in [41] as the optimal value. Since $1/9$, which is the actual contamination rate when the IR is 1:8 is a value closer to 0.03 as compared to $1/3$ (when IR $=$ 1:2), the Elliptic Envelope algorithm performs better in the former case. This signifies that for the Random Forest Feature Selection

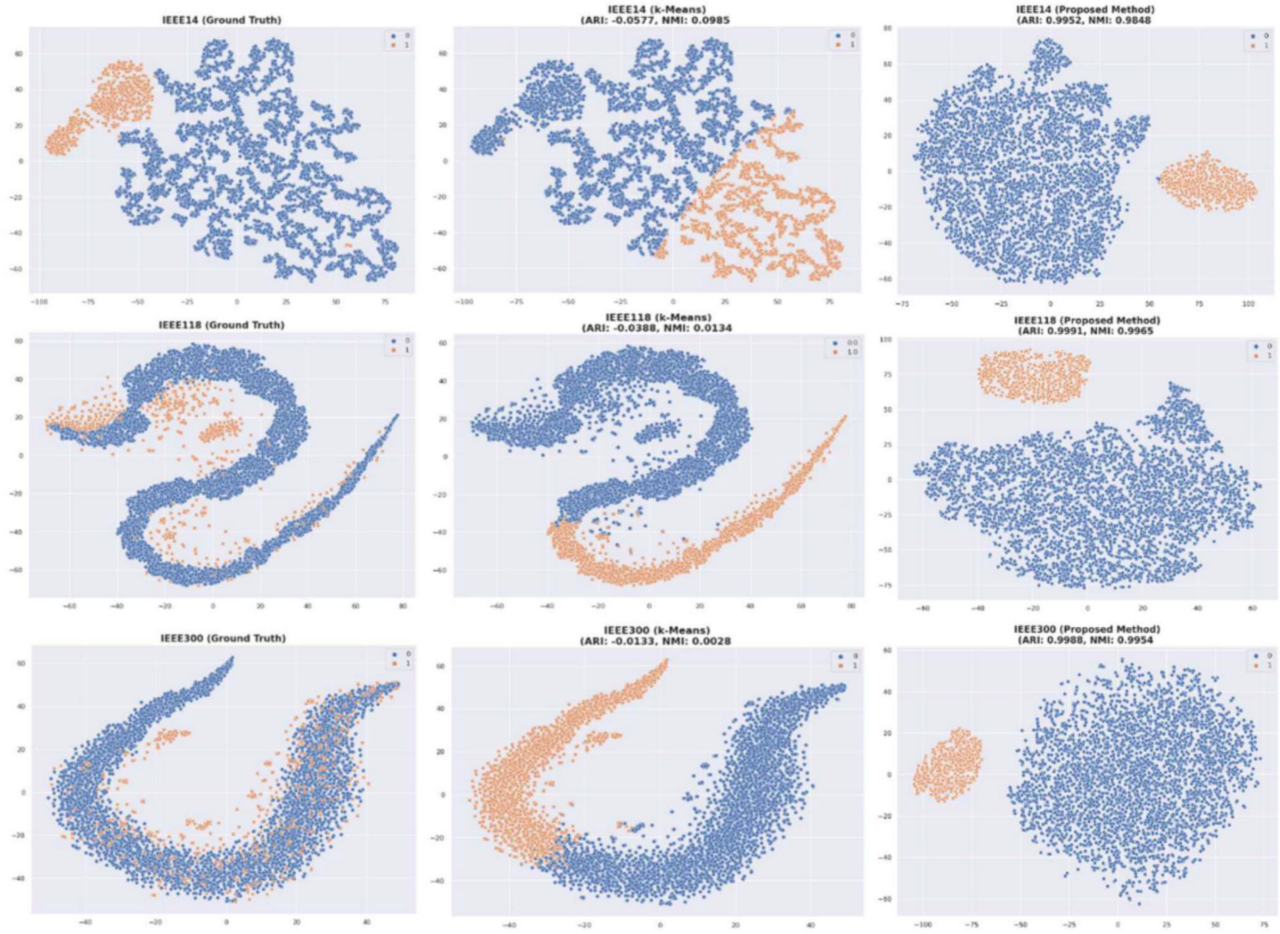


Fig. 7. Visualization of t-SNE projections of the ground truth data (1st column of each row), clusters obtained using k-Means baseline algorithm(2nd column of each row) and clusters obtained using our proposed method (3rd column of each row) when the imbalance ratio is approximately equal to 1:8. The three rows correspond to the IEEE 14 bus, 118 bus and 300 bus test systems respectively. The data is inseparable in the original feature space but forms clean clusters in the latent variable space.

+ Elliptic Envelope algorithm to perform satisfactorily, one has to exactly know the amount of anomaly in the dataset, which is impossible to know beforehand. A similar trend can be seen for the Isolation Forest based baseline [42]. The reason behind this is that Isolation Forests naturally work with an assumption that anomalous datapoints are few and significantly different from benign data. The PCA+DBSCAN [42] and the DAGMM [19] methods perform better than the other baselines. However, the performance of the DAGMM method drops when the attack model is changed from the data-driven model to the optimization-based model. The PCA+DBSCAN baseline closely follows the performance of the proposed method. However, both the DAGMM and PCA+DBSCAN algorithms use label information for supervision at some point in their training algorithm and hence are not completely unsupervised in nature. The performance of the proposed method, however, is consistent. In all the cases, it reaches an accuracy and F1 score of more than 99% consistently.

VI. Conclusion

The paper presented an unsupervised Deep Latent Space Clustering Algorithm for detecting stealthy FDIA in the Smart Grids. The process involves a sophisticated training procedure for a stacked autoencoder network using a greedy layer-wise training followed by an end-to-end finetuning of the stacked encoder-decoder system. Following this, the pretrained encoder network and a clustering head with trainable cluster centres is finetuned by minimizing a KL Divergence error between the soft cluster assignment outputs of the clustering head and an auxiliary target distribution. This self training procedure helps update the cluster centres and the encoder network weights to further improve the clustering performance. The non-linear transformation learned by the proposed method is superior to other supervised, semi-supervised and unsupervised baselines as far as the task of segregating corrupted and benign data samples are concerned. This is established by conducting rigorous experiments on the IEEE 14 bus, 118 bus and 300 bus test systems. Further corroborating evidence was presented by analysing visual maps generated by projecting the clustered data samples using the t-SNE algorithm.

References

[1] X. Liu, Z. Bao, D. Lu, and Z. Li, "Modeling of local false data injection attacks with reduced network information," *IEEE Trans. Smart Grid*, vol. 6, no. 4, pp. 1686–1696, Jul. 2015.

[2] O. Kosut, L. Jia, R. J. Thomas, and L. Tong, "Malicious data attacks on the smart grid," *IEEE Trans. Smart Grid*, vol. 2, no. 4, pp. 645–658, Dec. 2011.

[3] R. Tan, V. B. Krishna, D. K. Y. Yau, and Z. Kalbarczyk, "Integrity attacks on real-time pricing in electric power grids," *ACM Trans. Inf. Syst. Security*, vol. 18, no. 2, pp. 1–33, 2015.

[4] J. Tian, B. Wang, Z. Wang, K. Cao, J. Li, and M. Ozay, "Joint adversarial example and false data injection attacks for state estimation in power systems," *IEEE Trans. Cybern.*, early access, Nov. 19, 2021, doi: 10.1109/TCYB.2021.3125345.

[5] S. Xie, J. Yang, K. Xie, Y. Liu, and Z. He, "Low-sparsity unobservable attacks against smart grid: Attack exposure analysis and a data-driven attack scheme," *IEEE Access*, vol. 5, pp. 8183–8193, 2017.

[6] J. Kim, L. Tong, and R. J. Thomas, "Subspace methods for data attack on state estimation: A data driven approach," *IEEE Trans. Signal Process.*, vol. 63, no. 5, pp. 1102–1114, Mar. 2015.

[7] G.-Y. Yang and X.-J. Li, "Complete stealthiness false data injection attacks against dynamic state estimation in cyber-physical systems," *Inf. Sci.*, vol. 586, pp. 408–423, Mar. 2022.

[8] M. A. Rahman and H. Mohsenian-Rad, "False data injection attacks against nonlinear state estimation in smart power grids," in *Proc. IEEE Power Energy Soc. General Meeting*, 2013, pp. 1–5.

[9] G. Hug and J. A. Giampapa, "Vulnerability assessment of AC state estimation with respect to false data injection cyber-attacks," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1362–1370, Sep. 2012.

[10] J. Zhao, L. Mili, and M. Wang, "A generalized false data injection attacks against power system nonlinear state estimator and countermeasures," *IEEE Trans. Power Syst.*, vol. 33, no. 5, pp. 4868–4877, Sep. 2018.

[11] C. Liu, H. Liang, and T. Chen, "Network parameter coordinated false data injection attacks against power system AC state estimation," *IEEE Trans. Smart Grid*, vol. 12, no. 2, pp. 1626–1639, Mar. 2021.

[12] X. Liu and Z. Li, "False data attacks against AC state estimation with incomplete network information," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2239–2248, Sep. 2017.

[13] R. Deng and H. Liang, "False data injection attacks with limited susceptance information and new countermeasures in smart grid," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1619–1628, Mar. 2019.

[14] W.-L. Chin, C.-H. Lee, and T. Jiang, "Blind false data attacks against AC state estimation based on geometric approach in smart grid communications," *IEEE Trans. Smart Grid*, vol. 9, no. 6, pp. 6298–6306, Nov. 2018.

[15] M. Du, G. Pierrou, X. Wang, and M. Kassouf, "Targeted false data injection attacks against AC state estimation without network parameters," *IEEE Trans. Smart Grid*, vol. 12, no. 6, pp. 5349–5361, Nov. 2021.

[16] R. Jiao, G. Xun, X. Liu, and G. Yan, "A new AC false data injection attack method without network information," *IEEE Trans. Smart Grid*, vol. 12, no. 6, pp. 5280–5289, Nov. 2021.

[17] O. Boyaci, M. R. Narimani, K. R. Davis, M. Ismail, T. J. Overbye, and E. Serpedin, "Joint detection and localization of stealth false data injection attacks in smart grids using graph neural networks," *IEEE Trans. Smart Grid*, vol. 13, no. 1, pp. 807–819, Jan. 2022.

[18] T. Wu et al., "Extreme learning machine-based state reconstruction for automatic attack filtering in cyber physical power system," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 1892–1904, Mar. 2021.

[19] C. Chen et al., "Data-driven detection of stealthy false data injection attack against power system state estimation," *IEEE Trans. Ind. Informat.*, vol. 18, no. 12, pp. 8467–8476, Dec. 2022.

[20] A. S. Musleh, G. Chen, and Z. Y. Dong, "A survey on the detection algorithms for false data injection attacks in smart grids," *IEEE Trans. Smart Grid*, vol. 11, no. 3, pp. 2218–2234, May 2020.

[21] A. S. Musleh, H. M. Khalid, S. M. Muyeen, and A. Al-Durra, "A prediction algorithm to enhance grid resilience toward cyber attacks in WAMCS applications," *IEEE Syst. J.*, vol. 13, no. 1, pp. 710–719, Mar. 2019.

[22] J. Duan, W. Zeng, and M.-Y. Chow, "Resilient distributed DC optimal power flow against data integrity attack," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3543–3552, Jul. 2018.

[23] R. Moslemi, A. Mesbahi, and J. M. Velni, "A fast, decentralized covariance selection-based approach to detect cyber attacks in smart grids," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4930–4941, Sep. 2018.

[24] B. Li, T. Ding, C. Huang, J. Zhao, Y. Yang, and Y. Chen, "Detecting false data injection attacks against power system state estimation with fast go-decomposition approach," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 2892–2904, May 2019.

[25] T. R. B. Kushal, K. Lai, and M. S. Illindala, "Risk-based mitigation of load curtailment cyber attack using intelligent agents in a shipboard power system," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 4741–4750, Sep. 2019.

[26] M. Esmalifalak, L. Liu, N. Nguyen, R. Zheng, and Z. Han, "Detecting stealthy false data injection using machine learning in smart grid," *IEEE Syst. J.*, vol. 11, no. 3, pp. 1644–1652, Sep. 2017.

[27] S. A. Foroutan and F. R. Salmasi, "Detection of false data injection attacks against state estimation in smart grids based on a mixture Gaussian distribution learning method," *IET Cyber Phys. Syst. Theory Appl.*, vol. 2, no. 4, pp. 161–171, 2017.

[28] Y. Wang, M. M. Amin, J. Fu, and H. B. Moussa, "A novel data analytical approach for false data injection cyber-physical attack mitigation in smart grids," *IEEE Access*, vol. 5, pp. 26022–26033, 2017.

[29] D. Wang, X. Wang, Y. Zhang, and L. Jin, "Detection of power grid disturbances and cyber-attacks based on machine learning," *J. Inf. Security Appl.*, vol. 46, pp. 42–52, Jun. 2019.

[30] A. Jindal, A. Dua, K. Kaur, M. Singh, N. Kumar, and S. Mishra, "Decision tree and SVM-based data analytics for theft detection in smart grid," *IEEE Trans. Ind. Informat.*, vol. 12, no. 3, pp. 1005–1016, Jun. 2016.

[31] R. Punmiya and S. Choe, "Energy theft detection using gradient boosting theft detector with feature engineering-based preprocessing," *IEEE Trans. Smart Grid*, vol. 10, no. 2, pp. 2326–2329, Mar. 2019.

[32] S. McLaughlin, B. Holbert, A. Fawaz, R. Berthier, and S. Zonouz, "A multi-sensor energy theft detection framework for advanced metering infrastructures," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 7, pp. 1319–1330, Jul. 2013.

[33] A. Shefaei, M. Mohammadpourfard, B. Mohammadi-Ivatloo, and Y. Weng, "Revealing a new vulnerability of distributed state estimation: A data integrity attack and an unsupervised detection algorithm," *IEEE Trans. Control Netw. Syst.*, vol. 9, no. 2, pp. 706–718, Jun. 2022.

[34] G. Zhang, J. Li, O. Bamisile, D. Cai, W. Hu, and Q. Huang, "Spatio-temporal correlation-based false data injection attack detection using deep convolutional neural network," *IEEE Trans. Smart Grid*, vol. 13, no. 1, pp. 750–761, Jan. 2022.

[35] G. Fenza, M. Gallo, and V. Loia, "Drift-aware methodology for anomaly detection in smart grid," *IEEE Access*, vol. 7, pp. 9645–9657, 2019.

[36] A. Ayad, H. E. Z. Farag, A. Youssef, and E. F. El-Saadany, "Detection of false data injection attacks in smart grids using recurrent neural networks," in *Proc. IEEE Power Energy Soc. Innov. Smart Grid Technol. Conf. (ISGT)*, 2018, pp. 1–5.

[37] I. A. Khan, N. Moustafa, D. Pi, K. M. Sallam, A. Y. Zomaya, and B. Li, "A new explainable deep learning framework for cyber threat discovery in industrial IoT networks," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 11604–11613, Jul. 2022.

[38] M. M. N. Aboelwafa, K. G. Seddik, M. H. Eldefrawy, Y. Gadallah, and M. Gidlund, "A machine-learning-based technique for false data injection attacks detection in industrial IoT," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 8462–8471, Sep. 2020.

[39] Y. Zhang, J. Wang, and B. Chen, "Detecting false data injection attacks in smart grids: A semi-supervised deep learning approach," *IEEE Trans. Smart Grid*, vol. 12, no. 1, pp. 623–634, Jan. 2021.

[40] Y. He, G. J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2505–2516, Sep. 2017.

[41] M. Ashrafuzzaman, S. Das, A. A. Jillepalli, Y. Chakhchoukh, and F. T. Sheldon, "Elliptic envelope based detection of stealthy false data injection attacks in smart grid control systems," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, 2020, pp. 1131–1137.

[42] S. Ahmed, Y. Lee, S.-H. Hyun, and I. Koo, "Unsupervised machine learning-based detection of covert data integrity assault in smart grid networks utilizing isolation forest," *IEEE Trans. Inf. Forensics Security*, vol. 14, pp. 2765–2777, 2019.

[43] A. Parizad and C. Hatziadoniu, "Cyber-attack detection using principal component analysis and noisy clustering algorithms: A collaborative machine learning-based framework," *IEEE Trans. Smart Grid*, vol. 13, no. 6, pp. 4848–4861, Nov. 2022.

[44] O. Boyaci et al., "Graph neural networks based detection of stealth false data injection attacks in smart grids," *IEEE Syst. J.*, vol. 16, no. 2, pp. 2946–2957, Jun. 2022.

[45] Y. Pan, T. Yao, Y. Li, Y. Wang, C.-W. Ngo, and T. Mei, "Transferrable prototypical networks for unsupervised domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 2234–2242.

[46] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, 2008.

[47] S. Binna, S. R. Kuppannagari, D. Engel, and V. K. Prasanna, "Subset level detection of false data injection attacks in smart grids," in *Proc. IEEE Conf. Technol. Sustain. (SusTech)*, 2018, pp. 1–7.

Arnab Bhattacharjee (Graduate Student Member, IEEE) received the B.Tech. degree (Hons.) in electrical and electronics engineering with a minor in computer applications from NIT Tiruchirappalli in 2019. He is currently pursuing the Ph.D. degree with the University of Queensland–IIT Delhi Academy of Research. His current research interests include application of machine learning and artificial intelligence in power systems, electric vehicles, generative deep learning, and reinforcement learning-based control and optimization.

Arnab Kumar Mondal received the Bachelor of Engineering degree in electronics and telecommunication from Jadavpur University, India, in 2013. Then, he joined the Centre for Development of Telematics (C-DOT), New Delhi, where he served as a Research Engineer until July 2018. In C-DOT, he had the opportunity to participate in cutting-edge projects such as the Dense Wavelength Division Multiplexing and Packet Optical Transport Platform. He is currently pursuing the Ph.D. degree from IIT Delhi in July 2018 under the guidance of Prof. Prathosh AP and Prof. P. Singla. His research interests lie primarily within the field of deep generative models and applied deep learning. His Ph.D. thesis is supported by the Prime Minister's Research Fellows Scheme by Government of India.

Ashu Verma (Senior Member, IEEE) received the B.Tech. degree in electrical engineering from NIT Hamirpur in 2001, the M.Tech. degree in power systems from the Indian Institute of Technology (IIT) Delhi, New Delhi, India, in 2002, and the Ph.D. degree in transmission expansion planning from the Electrical Engineering Department, IIT Delhi in 2010, where she is currently an Associate Professor with the Centre for Energy Studies. Her current areas of research include power system planning, operation and control aspects of integrated renewable energy systems, smart buildings, and microgrids.

Sukumar Mishra (Senior Member, IEEE) received the M.Tech. and Ph.D. degrees in electrical engineering from the National Institute of Technology Rourkela, in 1992 and 2000, respectively.

He is currently a Professor with the Indian Institute of Technology Delhi and has been its part for the past 17 years. He has also been recognized as the INAE Industry Academic Distinguished Professor. His research interests lie in the field of power systems, power quality studies, renewable energy, and smart grid. He is currently acting as the ABB Chair Professor and has previously delegated as the NTPC, INAE, and Power Grid Chair Professor. He has also served as an Independent Director of the Cross Border Power Transmission Company Ltd., and the River Engineering Pvt. Ltd., and has carried out many important industrial consultations with TATA Power, Microtek, and others. He is the Founder of Silov Solutions Private Ltd., a company that specifically deals in products related to renewable energy sources utilizable at household scale as well as at commercial setups. Since March 2020, he has also been functioning as the Associate Dean of the Research and Development of IIT Delhi. He has won many accolades such as the INSA Medal for Young Scientist in 2002, the INAE Young Engineer Award in 2002 and 2009, the INAE Silver Jubilee Young Engineer Award in 2012, the Samanta Chandra Shekhar Award in 2016, the IETE Bimal Bose Award in 2019, the National Mission Innovation Champion Award in 2019, and the NASI-Reliance Industries Platinum Jubilee Award for Application Oriented Innovation in Physical Sciences in 2019. He has been granted fellowships from academies like NASI (India), INAE (India), and professional societies like IET (U.K.), IETE (India), IE (India). He has been working in close association with IEEE Delhi Section Executive Committee for the past few years and is currently serving as an Editor for the IEEE TRANSACTIONS ON SMART GRID, the IEEE TRANSACTIONS ON SUSTAINABLE ENERGY, and was an Area Editor for the *IET Generation, Transmission and Distribution*.

Tapan K. Saha (Fellow, IEEE) was born in Bangladesh and immigrated to Australia in 1989. He received the B.Sc. Engineering degree in electrical and electronic from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 1982, the M.Tech. degree in electrical engineering from the Indian Institute of Technology, New Delhi, India, in 1985, and the Ph.D. degree from the University of Queensland, Brisbane, Australia, in 1994. He is currently a Professor of Electrical Engineering with the School of Information Technology and Electrical Engineering, University of Queensland. His research interests include renewable energy integration to grid and condition monitoring of electrical plants. He is a Fellow with the Institution of Engineers, Australia.