A3D: Attention-based Auto-encoder Anomaly Detector for False Data Injection Attacks

Arnav Kundu, Abhijeet Sahu, Erchin Serpedin and Katherine Davis
Department of Electrical and Computer Engineering
Texas A&M University, College Station, TX
{arnav1993k, abhijeet_ntpc, eserpedin, katedavis}@tamu.edu

***Abstract*—With the influx of more advanced and more connected computing and control devices, the electric power grid has continuously evolved to rely on communication networks for efficient operation and control. A challenge with these new technologies is that they may introduce new and unforeseen avenues of access, making the grid more susceptible to cyber attacks. False Data Injection Attacks (FDIA) are a particular type of attack that aims to cause disruptions in the operation of the power grid by affecting the feedback mechanism to control the grid. This is carried out by modifying the measurements which enable a state estimator to approximate the state of the system. These attacks are designed in such a way that they preserve the system equations on which the state estimator operates; therefore, they cannot be detected by a simple residual-based detection mechanism. In this paper, we propose monotonic attention based auto-encoders, an unsupervised learning technique to detect FDIAs. The auto-encoder is trained under normal operating conditions, and we hypothesize that it will produce outputs which are close to the true system values at normal operation even if the measurements are modified by an adversary. Based on this hypothesis, that high reconstruction error occurs for the attacked conditions, the intrusion detection is performed by a threshold mechanism using Precision-Recall curve. We validate the efficacy of our proposed attention-based auto-encoder anomaly detector (A3D) over other variants of auto-encoders such as ANN and RNN based auto-encoders, and a few supervised learning techniques, by performing FDIAs on a IEEE 14 bus system.**

***Index Terms*—Anomaly Detection, Auto-Encoders, Monotonic Attention, False data Injection Attacks, Recurrent Neural Networks**

I. Introduction

The electric power grid is a piece of complex machinery involving multiple power generation sources, transmission lines, and distribution substations. The grid is also a cyber-physical critical infrastructure that supports most aspects of modern society. The grid requires continuous monitoring for safe and reliable performance. The electric power grid has evolved immensely over time, and the modern smart grid uses an integration of information and communication technologies (ICT) and supervisory control and data acquisition (SCADA) for efficient remote monitoring and real-time control. These ICTs rely on shared infrastructure, mainly because of the ease of access. Information can potentially be accessed, extracted, and modified by hackers. Such cyber-attacks occurred in countries like Ukraine [1], where an attacker opened circuit breakers to cause a power outage after intruding into the SCADA system. Other cyber-attacks like Pivnichna [2] caused a power outage, while Stuxnet [3] allowed control of programmable logic controllers (PLCs). Since the electric power grid is a large-scale and complex cyber-physical system, cyber-attacks can cause severe physical damages to the system.

False Data Injection Attacks (FDIAs) in a transmission system are a class of attacks that can trick the state estimator into predicting incorrect state values without being detected. One of the first references mentioning the impact of FDIAs on power systems was reported by Liu *et al.* [4]. State estimation bad data detection has traditionally used residual-based methods [5]. The challenge is that stealthy FDIAs cannot be detected using conventional residual-based methods since they are specifically designed to bypass these mechanisms [4, 6, 7]. Residual-based methods also do not consider the temporal structure of the measurement data. Such methods serve to detect severe measurement errors in state estimators, where bad data is non-malicious. The FDIAs of interest are specifically designed to ensure that bad data can be injected while keeping residual error negligible [4, 6].

For intruders to launch such an attack, it was previously thought that they need to have complete knowledge of the system, which would be difficult, but it has been proven that such attacks are possible even with localized partial information [6, 8]. The network can be protected from such attacks using two strategies: protection of critical measuring instruments and detection of attacks. Protection-based methods try to find an optimal set of measuring devices which need to be secured [9, 10]. Relying solely on device protection methods has limited practicality; the number of devices needed to be secured grows with the number of system states [11]. Furthermore, no security is perfect.

Detection-based methods try to find anomalies in the data that has been received [12–15]. Such methods depend on the real-time correlation between data points or the temporal structure of the data to classify a new set of measurements as anomalous. A significant drawback of using supervised anomaly detection is that it does not adapt well to changing patterns in transmission behavior over time [16, 17]. In [17], the authors propose a Gaussian mixture model to model the distribution of normal operating conditions and separate it from that of intrusion states.

Deep learning has shown significant promises in solving complex tasks. It has been used in pattern recognition problems like object detection [18], speech recognition [19], and anomaly detection [20]. Deep learning uses a data-driven approach where a function approximator is trained using

The material presented in this paper is based upon work supported by the NSF division of Electrical, Communication and Cyber System under Award Number 1808064.

gradient descent over a given set of data points [21]. The success of deep learning can be attributed to both the ability of neural networks to learn complex functions as well as the availability of massive data-sets. Motivated by its application and success in the field of speech recognition [19] and anomaly detection [20], this paper explores how deep neural networks can be applied to detect false data injection attacks in the electric power grid.

In this paper, we present the development of unsupervised methods using auto-encoders for the detection and location of FDIAs. We analyze the performance of a fully connected Artificial Neural Network (ANN) based auto-encoder, a simple Recurrent Neural Network (RNN) based auto-encoder, and finally an RNN auto-encoder with attention (A3D). We compare these unsupervised models with previously developed methods for supervised intrusion detection, including ANN-based detectors and RNN-based detectors for random levels of intrusion. We also compare our models with some previously used techniques like One-Class SVM [12] and Auto-Regressive Prediction [22]. Moreover, we present how auto-encoders can be employed to locate areas of intrusion. We then analyze the performance of the auto-encoders for the location of intrusions, and we finally summarize our observations and suggest future improvements.

II. Background

Since FDIAs are designed to bypass the bad data detection mechanism of the state estimator, several innovative approaches have been studied in the past for capturing the spatial and temporal correlations in the data to detect anomalous measurements. A traditional anomaly detection system can be designed as a prediction versus real data problem [22]. In [23], the authors propose to use independent data like forecast and historical patterns to detect anomalies. However, both of these methods depend on linear models; therefore, cannot be expected to operate correctly to capture non-linearities induced in complex AC state estimation [5]. They are also based on static thresholds which need to be carefully chosen and adapted over time.

In [24], a tree-pruning based approximation algorithm is used to detect anomalies in a graphical model. Inspired by various classical machine learning applications in cyber intrusion, sensor networks, and image processing, researchers have tried to apply nearest neighbor classifiers and other statistical classification techniques [25]. Ozay et al. [13] formulated this anomaly detection problem as a classification task where attacked and non-attacked scenarios are separately labeled in the training phase. The authors used methods like Sparse Logistic Regression, K-Nearest Neighbour (KNN), and Support Vector Machines (SVM) to classify a given set of measurements. It was found that KNN worked well for smaller grids (< 9 bus cases), while SVM with kernel worked well for larger power system cases (≥ 9 bus cases). It was also observed that finding correct hyper-parameters for SVM was difficult and computationally expensive. The authors generated training and test data, assuming a random number of devices were compromised. Therefore, the simple models perform well because the training dataset is similar to the test distribution. We believe generating such extensive training datasets for large power system cases is practically infeasible, and without such extensive training, these methods will fail. We have proven this with the results in Section V.

A protection approach based on power system characteristics is tested in [10], where a strategically selected set of devices are protected to make it impossible for an attacker to introduce an FDIA. FDIAs are possible with complete certainty only if certain devices are compromised [4]. The authors propose finding the measurement devices to be protected using a basic measurement identification method.

FDIAs can be detected by perturbing control parameters in the system, such as Distributed Flexible AC Transmission System (D-FACTS) devices that change effective line impedance [26], and comparing actual measurements with expected measurements [8, 27].

Another approach for anomaly detection is based on unsupervised methods where no labeled anomaly data is available. Most of these methods rely on density-based algorithms to find an approximate density cluster of non-anomalous data points. The points lying outside some margin of these density clusters are marked as anomalies [28]. One-Class SVM [29] is another example of unsupervised anomaly detection where the classifier is trained to know what is usual or non-anomalous. This helps to form a boundary region for general data, and anything lying outside those boundaries is flagged as an anomaly. However, since this is a linear classifier, it is sometimes challenging to find an optimal kernel which can construct the correct margins for this method.

Deep neural networks have been used in supervised [30], semi-supervised [31], and unsupervised settings [32] in the past for anomaly detection. Specifically, for anomaly detection in spatially and temporally correlated data, direct supervision using classification networks and unsupervised methods using auto-encoders have shown impressive results in the past. Applications of the same have been shown in the detection of electricity theft [33, 34].

Recurrent Neural Networks (RNNs) have been used in multiple time-series models involving neural networks like stock price predictions [35], language models [36]. RNNs have a memory component which can store previous inputs and outputs to predict the next state [21]. An RNN tries to map a sequence of inputs $x \in x_0, x_1, ..., x_t$ to a sequence of outputs $y \in y_0, y_1, ..., y_T$, where $T \geq 1$. Being a dynamic system, RNN-cells use a feedback mechanism to encode the temporal structure of the sequence so that subsequent outputs depend on current inputs. This is done using a hidden state at every time-step, which is considered while computing the output at the next state, as shown in Figure 1.

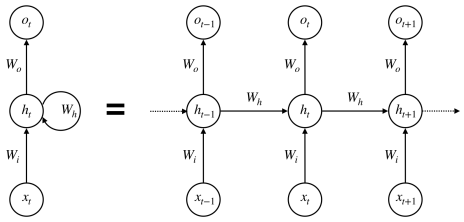

Fig. 1. Unrolled RNN-cell(W_i, W_o and W_h correspond to the input, output and hidden weights respectively and h represents the hidden state)

An *auto-encoder* is a type of neural network which helps to perform compression and restoration of its inputs; it is

comprised of an encoder and a decoder. The encoder compresses the input into a lower-dimensional latent state, and the decoder uses the latent state to predict the input back. In a classical sequence-to-sequence auto-encoder model [37], the encoder encodes the entire sequence in its hidden state at the last time step. This hidden state is then fed into a decoder to predict the input sequence. In many sequence-to-sequence learning problems, it was found that the encoded state was not enough for the decoder to predict the outputs, and an *attention mechanism* [38] can solve this problem. In a power system with its response governed by nonlinear differential and algebraic equations, it is logical to assume that there exists a monotonic temporal correlation, i.e., the past influences the future. Therefore, the model should be able to encode past measurements $(x_0, x_1, ...x_{t-1})$ with the required importance as present measurements (x_t). Attention allows us to draw a correlation between input and output sequences by weighing the contribution of every input sequence element to every output sequence element.

III. Detection using Unsupervised Techniques

Training supervised models for large power grids faces the challenge of generating attack labeled data. Therefore, relying on supervised learning methods might not be reasonable for large grids. We devise an auto-encoder based technique to distinguish normal operating conditions from attacked states. We have also explored how the same model can be employed to identify the affected region of the attacks.

A. Problem setting

As mentioned earlier, auto-encoders compress the network's inputs into a latent space and reproduce the inputs from the latent space. An auto-encoder is trained to minimize the error between its inputs and outputs, called the *reconstruction error*. We hypothesize that auto-encoders are useful in designing FDIA detection mechanisms in power systems. This hypothesis is based on the concept that an auto-encoder trained on data from normal operating conditions will predict outputs close to them even when subjected to malicious data. Power systems are generally over-determined to enable state estimators to predict the true state of the system. Moreover, the attack vectors are generally sparse [7]. Therefore, in the case of an attack, the data redundancy of the power system can be exploited by the neural network to predict outputs close to actual operating conditions. This will lead to a high reconstruction error, indicating an intrusion.

B. Model architectures

1) ***Fully-connected ANN auto-encoder****:* This model uses a fully-connected neural network for compressing and restoring its inputs. The model takes in measurements from a given time-step as inputs, compresses them into a latent space, and then expands them back to the input space, as shown in Figure 2. There are four layers: two encoders and two decoders. As shown in the figure, the model takes an input $Z_i \in \mathcal{R}^{m\times 1}$ and compresses it into a first latent space $E_i^1 \in \mathcal{R}^{h_1\times 1}$ for the i^{th} time-step (where m is the number of measurement devices on the grid, and h_1 is the cardinality of the first latent space). The weights of the first layer are denoted by $W_1 \in \mathcal{R}^{m\times h_1}$, which is followed by an Exponential Linear Unit (ELU) [39] non-linearity. The second encoder layer further projects E_i^1 into a lower-dimensional latent space E_i^2 using layer weights $W_2 \in \mathcal{R}^{h_1\times h_2}$ and ELU non-linearity. The decoder uses E_i^2 to reconstruct the inputs with the help of two layers with weights W_3 and W_4. The output of the third layer is passed through ELU function, and the fourth layer acts as a linear function which produces an output O_i. The process of encoding and decoding can also be done through two layers, but we have used a higher number of layers to allow a higher degree of freedom to the model for better approximation.

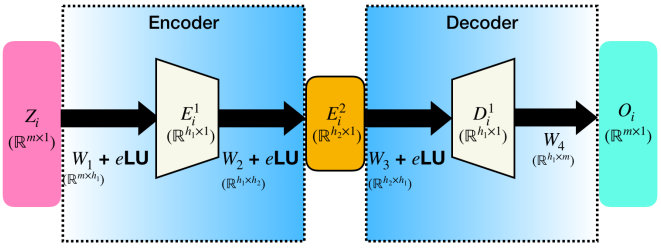

Fig. 2. Architecture of the fully connected ANN Auto-Encoder

We train this model on data from regular grid operating conditions to minimize the squared reconstruction error $(O_i - Z_i)^2$. Conventionally, over-fitting in a model is addressed using regularization, but, we do not consider regularization to ensure that the model over-fits on typical conditions and throws a high reconstruction error when subjected to intrusions. To correctly classify intrusions, we use an algorithm to fine-tune the reconstruction error threshold based on a few validation data as illustrated in III-C.

Attack vectors are generally sparse [4, 6, 7] and the reconstruction error for this model will be high only for cases with sparse attack vectors. If majority of the measurement devices are compromised then the reconstruction error will not be as high as expected. We can use a fully connected auto-encoder framework as used in this case, for measurements from a sequence of time-steps, which will force sparsity. The problem with this approach will be the number of parameters of the model as we increase the size of the sequence. Therefore, we use recurrent neural networks to solve this problem, and the model is explained in III-B2.

2) ***RNN auto-encoder****:* Just like the ANN auto-encoder, the RNN auto-encoder also has a set of encoder layers and decoder layers, as shown in Figure 3. To maintain consistency among the models, we have used the same number of encoder and decoder layers as used in ANN. The encoder comprises of two LSTM layers. The first layer with weights $W_1 = [W_i^1, W_h^1, W_o^1] \in \mathcal{R}^{m\times h_1}$ compress the input $Z \in \mathcal{R}^{m\times k}$ to $E_1 \in \mathcal{R}^{h_1\times k}$. The second layer uses E_1 as inputs and compresses it further to a condensed representation of all time steps $E_2 \in \mathcal{R}^{h_2\times 1}$. This means we use the last state of the second LSTM layer as the final encoded state (E_2). While decoding, we use E_2 as input to all time steps for the first decoder layer with weights $W_3 = [W_i^3, W_h^3, W_o^3] \in \mathcal{R}^{h_2\times h_1}$ to produce output $D_1 \in \mathcal{R}^{h_1\times k}$. D_1 is then upscaled back to $O \in \mathcal{R}^{m\times k}$ by the last LSTM-layer. The initial hidden states for all the LSTM-layers are allowed to be learned. We also used two different decoding mechanisms: front to back and back to front, among which we found the latter better. Back

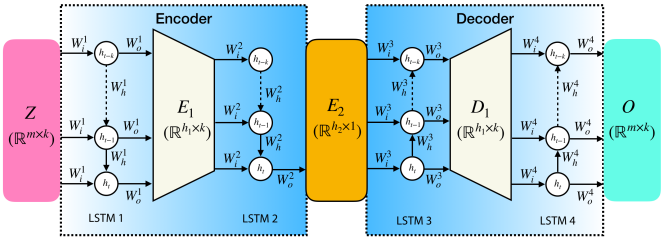

Fig. 3. Architecture of RNN Auto-Encoder

to front decoding means we decode the time-series in reverse order, i.e., the last input time-step is the first output time-step. Just like the previous case, this model is trained to minimize the reconstruction error of the sequence.

However, the RNN auto-encoder looses its performance (F1 Score) when the sequence length is long (>5 time-steps). This is because the compressed encoded state does not have enough space to represent all states in the entire sequence correctly [38]. Therefore, we propose the usage of attention based decoding [38] to ensure that there is minimal loss of information.

*3) **Attention-based Auto-encoder Anomaly Detector (A3D)**:* In the case of the RNN Auto-encoder, we were using the final output from the last encoded time-step as the input for the decoder. Instead, we use a weighted average of all encoded outputs from all time-steps. An *attention mechanism* [40] allows us to find the optimal weight of every encoder output for computing the decoder inputs at a given time-step, as shown in Figure 4. These weights are called *attention vectors* and are defined for every time-step. The attention vector is computed using the last hidden state, as shown in Figure 5. The h dimensional hidden state is multiplied with the attention matrix $(\mathcal{R}^{(h\times k)})$, which is passed through a softmax function to produce the attention vector $(\mathcal{R}^{(1\times k)})$. Here k stands for sequence length. The attention vector is then multiplied by the encoder outputs to produce a h dimensional input for the decoder LSTM cell.

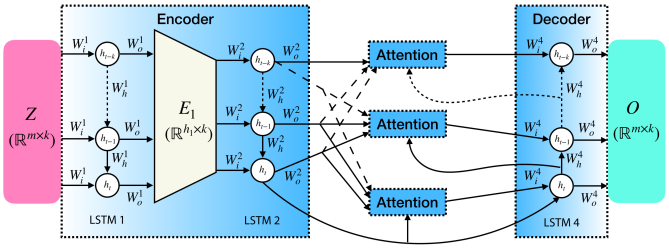

Fig. 4. Architecture of Auto-Encoder with Attention

In order to standardize our approach across all models, we use only one decoding layer with attention to keep the effective number of layers in the auto-encoder same. The model is trained to minimize the reconstruction loss of the sequence, and the training loss was found to be lower than that of simple RNN auto-encoder.

C. Thresholding and classification

An obvious question that arises while using the reconstruction error based on unsupervised methods mentioned above

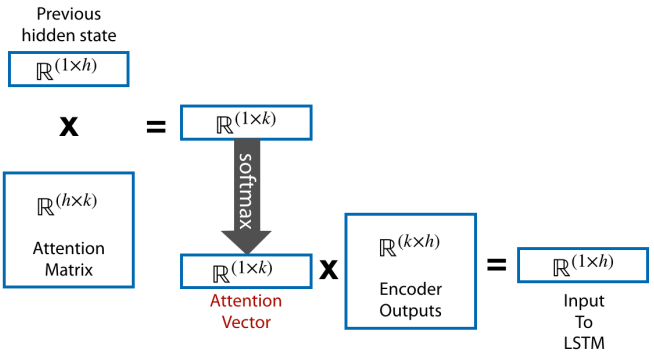

Fig. 5. Attention network

is: *How much error is allowed before raising an anomaly*? One possible method of deciding the threshold can be based statistical hypothesis testing [41], which uses the divergence of reconstruction error from training error to classify intrusions. But defining the significance level for this test can be challenging. To address this we use a small validation set to define the correct significance level for the test as explained in Algorithm 1.

Algorithm 1 Threshold mechanism for classifying intrusions

function THRESHOLD($valErr$, $meanErr$, $stdErr$, y)
  $div = \text{abs}(valErr - meanErr)/stdErr$
  $maxDiv = \max(div, \text{axis} = 1)$
  $sigDiv = \text{Sigmoid}(maxDiv)$
  $p, r, t = \text{PRCurve}(y, sigDiv)$
  for $precision, recall, thres$ in p, r, t **do**
    if $recall > 0.95$ and $precision > 0.80$ **then**
      $threshold = thres$
    end if
  end for
  return $threshold$
end function

We take the absolute difference between the reconstruction error of the validation set and mean training error and divide it by the standard deviation of training error to find divergence (div) at the device level. Next, we use a *Sigmoid* function on the device with maximum divergence to obtain $sigDiv$. Further, using precision-recall curve on $sigDiv$ and correct labels (y) of the validation data, we obtain a set of threshold values and the precision and recall corresponding to them. Since this is an anomaly detection problem we need a higher recall, therefore, we use a recall 0.95 along with a precision 0.8 for our experiments. These values can be tuned as per the requirement of the problem.

IV. Location of Intrusions using Auto-Encoders

The reconstruction error explained in the previous section can also be used for attack localization, to identify devices which are at compromised. As discussed earlier, we have used the maximum divergence to find thresholds for detecting intrusions. The same algorithm has been modified, and here instead of using maximum divergence, we obtain the precision-recall

curve and calculate classification thresholds for every device separately. During the evaluation, divergence is calculated at the device level, and the devices showing divergence more than their respective thresholds are considered compromised.

However, in an interconnected power system, measurements of a device is not independent of few other devices in its neighbor. This means a compromised device can affect the measurements on a different uncompromised device triggering a false alarm for that uncompromised device, resulting in high false positive and hence a low precision. Therefore, the approach of locating exact devices might not be the most optimal method of locating intrusions. We propose a connection-based device clustering approach to locate intrusions at the bus level. We hypothesize that any change due to FDIA on a particular device can have its immediate impact on its respective bus and the adjacent buses. Similarly, a group of connected devices can have spatio-temporal patterns like that of the entire grid. Therefore, if one device among the group of devices is found to be compromised, we classify the bus as compromised. For forming groups of connected devices, for a given bus we list:
1. All devices on that particular bus.
2. All devices on the lines connecting that bus to its adjacent buses.
This strategy can help to reduce the number of false positives, as all the possible devices where the effect of an attack might propagate during reconstruction, have been clustered.

V. Observations and Results

A. Data Generation

The data for normal operating conditions are first generated by using real-world power flow data from a portion of the Texas grid, and simulating its temporal patterns on an IEEE 14-bus case to capture all measurements. Next, we continued our simulation, and introduced a least-effort random attack that resulted in less residual error, preventing it from being detected by conventional bad data detection techniques, such as chi-square test, as demonstrated in our work [42]. We have used attack data for 3000 time-steps, where 1 time-step is 5 mins. For fixing the thresholds of classification, validation data are considered from the intrusion levels of 4, 12, and 20 devices being compromised out of the 39 measuring devices.

B. Evaluation metrics

We assume a lower probability of occurrence of intrusion in comparison to normal operating conditions. Since this is an intrusion detection problem, it is essential to have almost zero false negatives, while false positives can be allowed. Therefore, we aim to keep a high recall of 95%, and a precision of 80% for detection (Algorithm 1). As mentioned earlier, we expect a very low precision for location, and we have found that to be true. Therefore, we use a threshold that can lead to the best F1-Score (harmonic mean of precision and recall) neglecting the importance of precision and recall individually.

C. Results

Out of the 39 measurement variables in the 14-bus system, we have evaluated the performance of our model on five levels of intrusion where 4, 8, 12, 16, and 20 devices are randomly compromised. Our model is further evaluated for data with random levels of intrusion with a maximum number of compromised devices being less than 20. Since we assume that the probability of an intrusion is much less than that of normal operating conditions, we generate intrusions up to 5% of the time that stay active for a maximum of 10 time-steps. The results for the detection of intrusions over the entire grid are shown in Table I.

TABLE I
F1-Scores for Detection (N=number of devices compromised, TS= training set)

N	4	8	12	16	20	Random
KNN	TS	0.984	TS	1.0000	TS	0.5063
SVM (RBF)	TS	0.916	TS	0.999	TS	0.375
Sup ANN	TS	0.988	TS	1.0000	TS	0.704
Sup RNN	TS	0.971	TS	1.0000	TS	0.718
One-Cl SVM	0.277	0.235	0.178	0.076	0.017	0.108
AR Predictor	0.666	0.699	0.677	0.666	0.682	0.580
ANN AE	0.943	0.954	0.961	0.949	0.959	0.922
RNN AE	0.936	0.992	0.943	0.989	0.991	0.974
A3D	0.944	0.986	0.979	0.994	0.988	**0.984**

As a comparison, we have trained and tested some conventional unsupervised anomaly detection techniques like One-class SVM, Auto-regressive model (AR Predictor). Additionally, we have trained supervised ANN and RNN models on test data from cases where 4, 12, and 20 devices are compromised, and we subsequently evaluated them on cases where 8, 16, and a random number of devices are compromised.

It can be observed that both the linear predictors (One-Class SVM and AR Predictor) perform poorly against the neural network based auto-encoders. For ANN and RNN Auto-Encoders, it can be observed that they perform almost at par with A3D in cases of intrusions with a fixed number of compromised devices. However, when subjected to random levels of intrusion, the ANN auto-encoder lags in performance.

A3D also outperforms the RNN auto-encoder in the case of random levels of intrusion. It can also be observed that the performance is poor when few devices are compromised. This might indicate that the changes introduced by the attacker are not significant to cause a divergence in the system behavior and can be ignored as random noise. We can also observe that the auto-encoder based models outperform the supervised models when a random number of devices are compromised. This might be because the amount of training data available to the supervised methods is not sufficient to cover all levels of intrusion. This might also be the reason for their exceptional performance in some cases (8, 16 devices). The RNN auto-encoder and A3D perform very closely to the supervised methods in the case of 8 and 16 devices compromised. Therefore, our hypothesis regarding a high reconstruction error for intruded states is true, and this approach efficiently covers all cases of intrusions. A3D turns out to be a consistent solution than both supervised and previously developed unsupervised techniques. Another point that can be noted is the performance of ANN auto-encoder with increase in number of devices compromised. We can observe that the performance starts to degrade once more than 12 devices are compromised which can validate our claim regarding the effect of sparsity of the attack.

As mentioned earlier, we have modified the thresholding mechanism to compute thresholds per device. We have extended our validation data and included cases with 8 and 16

TABLE II
F1-SCORES FOR LOCATION AT DEVICE LEVEL

N	4	8	12	16	20	Random
ANN AE	0.768	0.837	0.895	0.904	0.929	0.265
RNN AE	0.451	0.608	0.759	0.706	0.758	0.440
A3D	0.650	0.763	0.798	0.753	0.764	**0.459**

TABLE III
F1-SCORES FOR LOCATION AT BUS LEVEL

N	4	8	12	16	20	Random
ANN AE	0.758	0.848	0.899	0.912	0.925	0.505
RNN AE	0.735	0.781	0.822	0.814	0.823	0.508
A3D	0.675	0.789	0.843	0.818	0.833	**0.544**

devices compromised. The results of this approach are shown in Table II. As expected, the F1-Scores are relatively low, and the main reason behind this is a low precision score due to distribution of reconstruction error among directly connected meters raising more false positives. During compression, the attack introduced on various meters gets combined into a dense representation. This leads to a high total divergence from the regular operation. However, during expansion the degree of freedom enjoyed by the change induced by the attack in the encoded space, is directly proportional to the number of outputs in the decoded state. Therefore, it is likely that the effect of the attack introduced gets distributed among the outputs. This is likely to cause high reconstruction error for most of the reconstructed outputs. This has been shown in Figure 6. The results also indicate that the performance of the ANN auto-encoder is inferior to that of RNN auto-encoder and A3D. This might be because both of them rely on temporal information in addition to spatial structure. It can also be observed that with increasing number of devices compromised the F1-Score improves. This is because of the increasing precision as the number of devices compromised increases. For location at the bus level, we have clustered

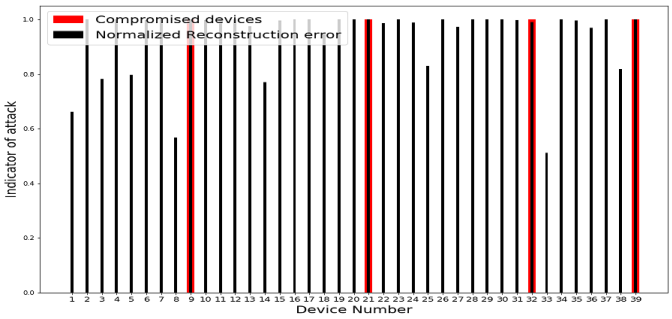

Fig. 6. Reconstruction error and true compromised devices

devices, as explained in Section IV. For a given device, we can expect the reconstruction function to be a function of the measurements from its adjacent devices. Therefore, for an attacked device, the change induced by the attack has a higher probability of flowing into the measurements adjacent to it during reconstruction. This can be seen by comparing Figure 6 and 7. From Figure 8 it can be seen that devices 9, 21, 32 and 39 are connected to buses 1, 2, 3, 9 and 14, and it can be seen that they get detected correctly ($Precision = 0.6,\ Recall = 1$). In addition, some false positives are also raised here, but it is lesser than in the case of locating exact devices. From Figure 8 it can also be seen that the devices affected by this attack are $1, 2, 4, 7, 8, 9, 11, 12, 15, 16, 17, 20, 21, 23, 26, 28, 29, 32, 33, 34, 35, 39$ (22 devices), which is very similar to the output seen in Figure 6. The results of the location at the bus level is shown in Table III. It can be observed that the performance of all the three auto-encoder models is approximately the same. This also validates the claim that the reconstruction error of a particular device depends on the measurements of its adjacent devices. Since the ANN auto-encoder relies completely on spatial information, it can be concluded that the function learnt by the ANN for a particular device depends primarily on the inputs from its adjacent devices. Therefore, an attack on a particular device affects the output received from the auto-encoder for its adjacent devices as well.

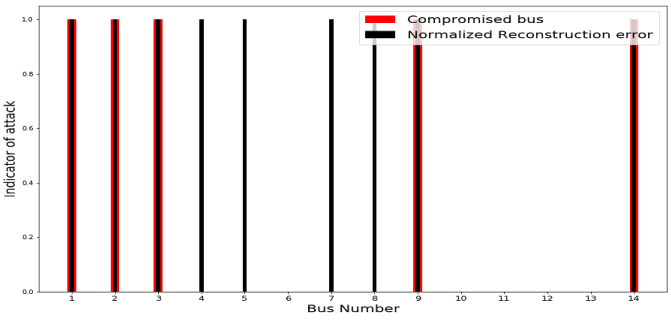

Fig. 7. Reconstruction error and true compromised busses

VI. CONCLUSION AND FUTURE WORK

We have demonstrated how auto-encoders can be useful for detecting FDIAs. Specifically, our improvised monotonic attention based auto-encoder performed better in comparison to ANN and RNN based auto-encoders for random levels of intrusions. Being an unsupervised learning technique it will not be needing a diverse training dataset to train complex neural network models. The idea is also coherent with the concept of state estimation because auto-encoders are generally used in noise filtering applications. In normal state estimation of a power system, we have a fixed number of state variables and a single time-step process. Here, we are considering an extended version of the same idea with multiple time-steps and a lower dimensional hidden state than the number of state variables.

Future work includes further testing of these models on larger cases like 118 bus or 300 bus systems. Additionally, they are yet to be trained and tested on contingencies and varying levels of renewable energy penetration, which would be important to consider.

REFERENCES

[1] D. Alert, "Analysis of the cyber attack on the ukrainian power grid," 2016.
[2] L. Streltsov, "The system of cybersecurity in ukraine: principles, actors, challenges, accomplishments," *European Journal for Security Research*, vol. 2, no. 2, pp. 147–184, 2017.
[3] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.
[4] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Transactions on Information and System Security*, vol. 14, no. 1, pp. 1–33, May 2011. [Online]. Available: https://doi.org/10.1145/1952982.1952995

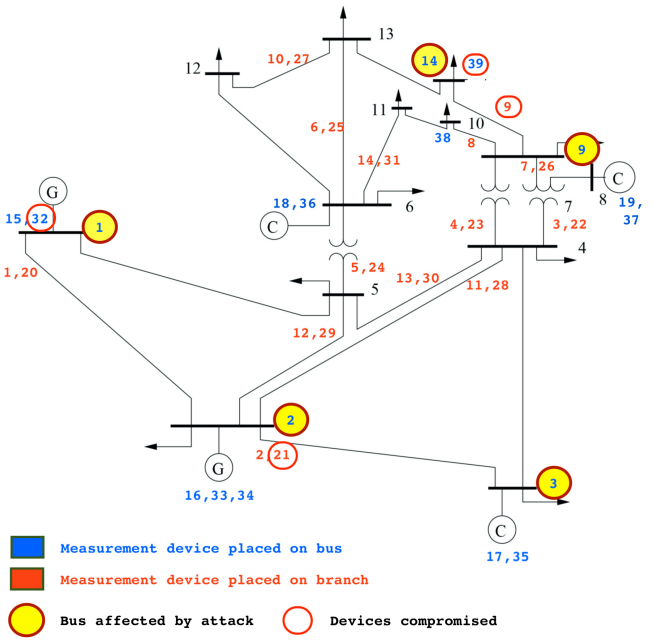

Fig. 8. 14 bus system along with placement of measurement devices and attack

[5] A. Monticelli, *State estimation in electric power systems: a generalized approach*. Springer Science & Business Media, 2012.

[6] X. Liu and Z. Li, "Local topology attacks in smart grids," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2617–2626, 2017.

[7] L. Liu, M. Esmalifalak, Q. Ding, V. A. Emesih, and Z. Han, "Detecting false data injection attacks on power grid by sparse optimization," *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 612–621, 2014.

[8] K. R. Davis, K. L. Morrow, R. Bobba, and E. Heine, "Power flow cyber attacks and perturbation-based defense," in *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2012, pp. 342–347.

[9] G. Chaojun, P. Jirutitijaroen, and M. Motani, "Detecting false data injection attacks in ac state estimation," *IEEE Transactions on Smart Grid*, vol. 6, no. 5, pp. 2476–2483, 2015.

[10] R. B. Bobba, K. M. Rogers, Q. Wang, H. Khurana, K. Nahrstedt, and T. J. Overbye, "Detecting false data injection attacks on dc state estimation," in *Preprints of the First Workshop on Secure Control Systems, CPSWEEK*, vol. 2010, 2010.

[11] Y. Zhou and Z. Miao, "Cyber attacks, detection and protection in smart grid state estimation," in *2016 North American Power Symposium (NAPS)*. IEEE, 2016, pp. 1–6.

[12] M. Ozay, I. Esnaola, F. T. Y. Vural, S. R. Kulkarni, and H. V. Poor, "Machine learning methods for attack detection in the smart grid," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 8, pp. 1773–1786, 2016.

[13] ——, "Smarter security in the smart grid," in *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2012, pp. 312–317.

[14] A. Ayad, H. E. Farag, A. Youssef, and E. 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)*. IEEE, 2018, pp. 1–5.

[15] 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 Transactions on Smart Grid*, vol. 8, no. 5, 2017.

[16] Y. Wang, W. Shi, Q. Jin, and J. Ma, "An accurate false data detection in smart grid based on residual recurrent neural network and adaptive threshold," in *2019 IEEE International Conference on Energy Internet (ICEI)*. IEEE, 2019, pp. 499–504.

[17] 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-Physical Systems: Theory & Applications*, vol. 2, no. 4, pp. 161–171, 2017.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[19] J. Li, V. Lavrukhin, B. Ginsburg, R. Leary, O. Kuchaiev, J. M. Cohen, H. Nguyen, and R. T. Gadde, "Jasper: An end-to-end convolutional neural acoustic model," *arXiv preprint arXiv:1904.03288*, 2019.

[20] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[21] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[22] A. H. Yaacob, I. K. Tan, S. F. Chien, and H. K. Tan, "Arima based network anomaly detection," in *2010 Second International Conference on Communication Software and Networks*. IEEE, 2010, pp. 205–209.

[23] A. Ashok, M. Govindarasu, and J. Wang, "Cyber-physical attack-resilient wide-area monitoring, protection, and control for the power grid," *Proceedings of the IEEE*, vol. 105, no. 7, pp. 1389–1407, 2017.

[24] S. Bi and Y. J. Zhang, "Graphical methods for defense against false-data injection attacks on power system state estimation," *IEEE Transactions on Smart Grid*, vol. 5, no. 3, pp. 1216–1227, 2014.

[25] P.-Y. Chen, S.-M. Cheng, and K.-C. Chen, "Smart attacks in smart grid communication networks," *IEEE Communications Magazine*, vol. 50, no. 8, pp. 24–29, 2012.

[26] D. Divan and H. Johal, "Distributed facts-a new concept for realizing grid power flow control," in *2005 IEEE 36th Power Electronics Specialists Conference*. IEEE, 2005, pp. 8–14.

[27] K. L. Morrow, E. Heine, K. M. Rogers, R. B. Bobba, and T. J. Overbye, "Topology perturbation for detecting malicious data injection," in *2012 45th Hawaii International Conference on System Sciences*. IEEE, 2012, pp. 2104–2113.

[28] U. S. Goldstein M, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PLoS ONE*, vol. 11, no. 4, 2016.

[29] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[30] R. Chalapathy, E. Z. Borzeshi, and M. Piccardi, "An investigation of recurrent neural architectures for drug name recognition," *arXiv preprint arXiv:1609.07585*, 2016.

[31] D. Wulsin, J. Blanco, R. Mani, and B. Litt, "Semi-supervised anomaly detection for eeg waveforms using deep belief nets," *2010 Ninth International Conference on Machine Learning and Applications*, 2010.

[32] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, "Deep learning for unsupervised insider threat detection in structured cybersecurity data streams," in *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[33] M. Ismail, M. Shahin, M. Shaaban, E. Serpedin, and K. Qaraqe, "Efficient detection of electricity theft cyber attacks in ami networks," in *2018 IEEE Wireless Communications and Networking Conference, WCNC 2018*, vol. 2018-April. Institute of Electrical and Electronics Engineers Inc., 6 2018, pp. 1–6.

[34] M. Nabil, M. Ismail, M. Mahmoud, M. Shahin, K. Qaraqe, and E. Serpedin, "Deep recurrent electricity theft detection in ami networks with random tuning of hyper-parameters," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 740–745.

[35] S. Selvin, R. Vinayakumar, E. Gopalakrishnan, V. K. Menon, and K. Soman, "Stock price prediction using lstm, rnn and cnn-sliding window model," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2017, pp. 1643–1647.

[36] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh annual conference of the international speech communication association*, 2010.

[37] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," in *Advances in Neural Information Processing Systems*, 2015, pp. 3079–3087.

[38] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[39] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[40] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015.

[41] J. Neyman and E. S. Pearson, "Ix. on the problem of the most efficient tests of statistical hypotheses," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 231, no. 694-706, pp. 289–337, 1933.

[42] A. Kundu, A. Sahu, K. Davis, and E. Serpedin, "Learning-based defense of false data injection attacks in power system state estimation," in *North American Power Symposium 2019*.