False Data Injection Attack Detection in Smart Grid State Estimation Using State Vector Residual Analysis and Random Forest Classification

K Durga Syam Prasad
Professor
Vignan's Institute of Information Technology, Duvvada, Visakhapatnam, Andhra Pradesh-530049
kdsyamprasad@vignaniit.edu.in

Sampath Kumar S,
Assistant professor, Department Artificial Intelligence & Data Science,
Don Bosco Institute of Technology, Bangalore, Karnataka
sampath.ksit@gmail.com

Arnab Ghosh, *Senior Member, IEEE*
Department of Electrical Engineering
National Institute of Technology
Rourkela, India

Ranjith Kumar Peddi
Principal Software Engineer,
Equinix Inc, Frisco, Texas, United States,
peddi.ranjithk@gmail.com,
ORCID ID: 0009-0003-4223-8578

S. Sri Nandhini Kowsalya
Assistant Professor, Department of Information Technology,
Vel Tech Multi Tech Dr. Rangarajan Dr. Sakunthala Engineering College,
Chennai, Tamil Nadu, India
srinandhinikowsalya@gmail.com

Syed Riyaz Ahammed
Associate Professor, Department of Electronics & Communication Engineering
NMAM Institute of Technology
NITTE (Deemed to be University),
Karnataka, India
Syedriyaz.global@gmail.com

***Abstract*— Smart grid controllers trust the sensors that are attached to their grids. By silently spoofing those sensor readings — keeping the corrupted values within normal-looking limits — the state estimator neither sees nor alerts danger, and the grid responds to a fake picture of itself. The False Data Injection problem, and standard chi-squared detectors are oblivious to it. This paper has a different method. We use the differences between what sensors report and what a rolling estimate based on the previous 30 minutes+ of readings says they should report to build features, rather than using raw measurements directly as inputs for classifiers. A 27-D feature space (raw readings, rolling residual and volatility metrics combined) obtained from the eight sensor channels of the real Mafate isolated microgrid on Réunion Island. We inject four types of attacks (scaling, bias, ramp, and replay) on real-measurement copies. Related Articles Here is how smart grid controllers trust sensors that connect to their grids. The static enemy is another high-tech tool: by pretending that the sensor readings are non-detectable — with corrupted values below an undetection, within normal-looking parameters — the state estimator sees no danger, and the grid reacts to a false representation of itself. So the False Data Injection problem that we are trained on (and also standard chi-squared detectors are blind to). This paper does it differently. Instead of feeding raw measurements to classifiers directly, we build features based on the differences between what sensors report and what an estimate calculated from previous readings (typically 30 minutes+) says they should be reporting. A 27-D feature space (raw readings, rolling residual and volatility metrics combined) extracted from the eight sensor channels of the real Mafate isolated microgrid on Réunion Island. We inject four attacks (scaling, bias, ramp and replay) on real-measurements copies.**

I. INTRODUCTION

The smart gControllers act on sensor data. Each dispatch decision, each voltage correction, each protection response is powered by measurements streaming in from the field. So, when an attacker captures that data and alters the numbers before they ever reach their destination, the controller reacts to a nonexistent grid. Internally consistent readings and nothing flagged as wrong by the state estimator [1], [2].

If you know the grid topology, constructing an injection vector that survives the residual checks is not difficult. Gene logic has long been just that — know the math and craft it to yield a consistency vector whereby its contribution satisfies the weighted least squares condition for bad-data detection, and waves your limited noise and run-of-the-mill chi-squared son will wave shoo into town [3], [4]. From there the possibilities are endless. Reporting increased load to entice dispatch into overscheduling generation. Add in a fixed offset to battery current and deplete reserves faster than the controller thinks it is. Increase a voltage measuring in steps to some edge the driver is unaware has arrived. Or, feed real history into the current stream — values look perfectly fine because this is exactly how history played out from the beginning of time; just not now [5].

None of the four is caught by a fixed threshold. Gradual ramps fall short of what a rule needs that is calibrated for scaling. However, a rule that is sensitive to detect ramps raises alarms during non-malicious transients. There is a need for something that learns using data for the detection problem [6]. Two detection families exist. Model-based approaches compare measurements to the relevant power flow equations and Kirchhoff's laws [7]. This sounds good in theory, but brittle if topology changes or the adversary corrupts the deterministic model first. Data-driven classifiers learn normal patterns from previously recorded data and indicate deviations [8]. The problem is that nearly all published studies test the IEEE 57-bus or the 118-bus synthetic systems. No PV intermittency, No battery variability, and No demand variability within one household. Increasing data on clean test buses does not translate into real deployments [9]

There is also a feature engineering gap. Most work feeds raw measurement vectors to the classifier. The residual — the gap between what a sensor reports and what a rolling-window estimate says it should report — carries more discriminative information about whether an attack is happening. That gap is underused [10]. This paper uses real data from the Mafate isolated microgrid on Réunion Island [13]. A 27-dimensional feature space built from raw measurements, rolling estimation residuals, and volatility metrics is used to train and compare three classifiers — Random Forest, Calibrated Linear SVM, and MLP — on a 166,305-sample dataset covering four attack types. Four contributions: (i) residual-based state vector feature

engineering on real microgrid data; (ii) systematic injection of four physically distinct attack modalities on real measurement copies; (iii) controlled three-classifier comparison; (iv) honest reporting of 48.25% replay detection — the open problem this work identifies.

DI theory is solid. An attacker with topology knowledge constructs injection vectors that pass the chi-squared test by design [3]. Replay-style attacks that preserve measurement distributions are hardest to catch [9]. Defence and recovery methods have been studied [9], but virtually all of this work uses IEEE test systems under clean conditions — not real microgrids with variable solar generation and multi-household demand. When the accuracy of the network model is guaranteed, locality detection can exploit graph-structured residual monitoring by using model-based detectors to localise attacked measurements [7], [11]. Simulation Characterization of PMU-Augmented Approaches Since the same PMU CAV simulation can achieve high specificity Neither is suitable for isolated microgrids where the topology may be simple, but generation is intermittent and system model drifts continuously.

In terms of approaches based on data, deep learning and ensemble methods have high performance on labelled injection datasets [8], [12]. LSTM networks capture temporal patterns that may be missed by single-timestep classifiers – as the duration of a sequence extends over several observations, ramp and replay attacks become detectable. Multi-node SCADA attack detection with graph neural networks [11] And the one constant limitation remains synthetic data. Test bus benchmarks: Models trained on the test bus benchmarks degrade on real operational measurements by including PV dynamics and battery cycles [6].

The second gap is the choice of the feature space. For many attacks that produce large instantaneous (or near-instantaneous) changes, using the raw measurements works well. Rolling residuals (the deviation of each channel from its local30-minute trend) pick up on attacks, which raw measurements average out within normal variation. This has been done in a few papers [10], but none based on realistic isolated microgrid data. Both gaps are addressed here.

II. METHODOLOGY

2.1 System Overview and Problem Formulation

The Mafate microgrid rests in a mountain valley in Réunion Island and are off grid. Everything is powered by 2 PV arrays, a lead-acid battery bank and three households. We receive sensor readings every minute across eight channels. By intercepting the communication line between sensors and the control unit, an attacker can replace real readings with fake ones before the energy management system ever sees them.

The task is simple classification — classify every observation as normal (0) or under attack. The tricky part is the features we give the classifier. Simply the raw measurements are not enough. The difference between the current reading of a sensor and what it recorded as an average over the last 30 minutes — that delta reacts quite differently under attack than in regular operation. The difference you point out formed the basis of this approach.

2.2 Dataset and Preprocessing

The Mafate instrumentation from the PIMENT Laboratory provides two measurement streams. The solar plant stream includes inverter output, both PV regulator outputs, battery voltage, current and temperature. The appliance-level and aggregate load demand stream is carried in the House 1 demand stream. This leaves a total of 118,791 clean measurement pairs after merging on timestamp and dropping rows with missing or non-numeric values.

Eight channels make up the state vector: P_pv1, P_pv2, P_out, U_pv1, U_pv2, U_bat, I_bat, and P_load. These eight cover everything the control system needs to know about the microgrid's operational state at any given minute.

2.3 State Vector Feature Engineering

Group 1—Raw and derived measurements (11 features). A total of eight state channels and 3 physical relationship factors. Power Balance of PVs: P_pv1 + P_pv2 − P_out. Battery voltage ratio (U_bat)/(U_pv1+ U_pv2) Power-to-load ratio: P_out per watts (scaled) divided by P_load. Now, the reason why these three derived terms are meaningful is that if you attack one channel it now breaks the physical relationship between channels right

Group 2 — Rolling residuals (8 features). Use the rolling mean over 30min as a pseudo state estimate for each channel Take this reading, subtract it. That difference is the residual. Under normal conditions, it hovers around the zero mark. It is moving under an biased assault. In a scaling attack the whole of the distribution stretches. It is trending under a ramp attack. Eight channels give eight residuals.

Group 3 — Rolling volatility (8 features). The 30-minute rolling standard deviation per channel. Ramp attacks produce a volatility signature that climbs monotonically as the perturbation grows. Replay attacks swap in real historical values — but from a different operational period, so the volatility pattern shifts in a way that differs from the current window's normal behaviour.D. Attack Simulation.

2.4 Generating Attack Data

The 118,791 real records become the normal class. 47,514 synthetic attack records are added across four modalities — all applied to copies of real measurements, not fabricated from scratch.

Scaling: Multiply 1–3 randomly selected channels by a factor drawn from U(1.3, 2.5) for amplification or U(0.3, 0.7) for attenuation. Direction chosen randomly per sample.

Bias: Add $\pm c \cdot \sigma_k$ to target channels, where $c \sim U(2.0, 5.0)$ and σ_k is the population standard deviation of that channel across the clean dataset.

Ramp: Apply a linearly growing perturbation from zero to $3\sigma_k$ across the attack window. The direction is random. The result is a measurement that drifts steadily away from truth over the window.

Replay: Replace current readings with values pulled from a window shifted by δ samples, where $\delta \sim U(500, 5000)$. The substituted values are real operational data — just from the wrong time period.

All attacks hit 1–3 channels per sample. Seeds fixed at 42 everywhere.

2.5 Classifier Design and Evaluation Protocol

Three classifiers run under identical conditions on an 80/20 stratified split — same features, same standardization fitted on training data only, same evaluation metrics.

Random Forest (RF): 200 trees, max depth of 20, min 5 samples per split and balanced class weights. This serves as the major classifier — it works well on high-dimensional tabular features, does not easily overfit, and provides native feature importance stats.

Calibrated Linear Support Vector Classifier (SVM): linear kernel, C = 1.0, balanced weights and Platt scaling for output probabilities For computational feasibility on the 133,044 training samples available, the linear kernel was selected

Multi-Layer Perceptron (MLP): Three hidden layers [128, 64, 32], ReLU activations, maximum 300 epochs, early stopping on validation loss. This allows checking that test results hold over different data splits through five-fold stratified cross-validation on the RF training partition

III. Mathematical Modeling

3.1 Derived Physical Features

Three physical interaction terms augment the eight raw state channels. PV power balance::

$$\Delta P_{pv}(t) = P_{pv1}(t) + P_{pv2}(t) - P_{out}(t) \quad (1)$$

Under normal operation this stays bounded by battery charging losses. Any single-channel FDI attack on PV or inverter measurements breaks this balance. Battery voltage ratio:

$$\rho_{bat}(t) = \frac{U_{bat}(t)}{U_{pv1}(t) + U_{pv2}(t) + \varepsilon} \quad (2)$$

This encodes the coupling between battery and PV bus voltages — voltage-targeting attacks disrupt it immediately. Power-to-load ratio.

$$\gamma_{pl}(t) = \frac{P_{out}(t) \times 10^3}{P_{load}(t) + \varepsilon} \quad (3)$$

Supply-side injection while load measurements remain clean produces a systematic deviation in this ratio, even when individual channel values appear plausible.

3.2 Rolling Estimation Residuals

For each channel k, a 30-sample causal rolling mean serves as a pseudo state estimate. The residual

$$r_k(t) = z_k(t) - \hat{z}_k(t) \quad (4)$$

follows a zero-mean distribution under normal operation. Under attack: a bias attack of magnitude b_k shifts $E[r_k(t)]$ by b_k; a scaling attack by factor α_k scales both mean and variance; a ramp attack introduces a linear trend in $r_k(t)$ over the attack window.

3.3 Rolling Volatility

The 30-sample rolling standard deviation per channel:

$$\sigma_k(t) = \sqrt{\frac{1}{W-1} \sum_{\tau=t-W+1}^{t} \left(z_k(\tau) - \hat{z}_k(t)\right)^2} \quad (5)$$

is elevated during ramp attacks (monotonically growing deviations within the window) and shifts pattern during replay attacks (historically-period values substituted for current-period measurements).

3.4 Complete Feature Vector

The full 27-dimensional feature vector submitted to the classifier is assembled as:

$$\phi(t) = \left[\mathbf{z}(t)^\top,\ \Delta P_{pv}(t),\ \rho_{bat}(t),\ \gamma_{pl}(t),\ r_1(t), \ldots, r_8(t),\ \sigma_1(t), \ldots, \sigma_8(t)\right]^\top \quad (6)$$

Group 1 (11 features): raw channels + 3 derived physical terms. Group 2 (8 features): rolling residuals. Group 3 (8 features): rolling volatility.

3.5 Attack Models

Scaling attack:

$$z'_k(t) = \alpha_k \cdot z_k(t) \quad (7)$$

Bias attack:

$$z'_k(t) = z_k(t) + b_k \quad (8)$$

Ramp attack:

$$z'_k(t_i) = z_k(t_i) + \frac{i}{n} \cdot 3\hat{\sigma}_k \cdot d_k, \quad i = 0,1, \ldots, n-1 \quad (9)$$

Replay attack:

$$z'_k(t) = z_k(t + \delta) \quad (10)$$

Scaling and bias produce immediate residual shifts; ramp builds gradually within the rolling window; replay is the hardest because it substitutes real historical measurements — preserving local statistics while replacing the current operational context

IV. Results and Discussions

The experimental results using the real operational microgrid data are presented in this section to assess the performance of the proposed AI-driven FDI attack detection framework. The evaluation framework consists of the characteristics from input data, the comparison of model performance on three AI classifiers as well as per attack type detection analysis and computational efficiency. All numerical values mentioned are directly traceable to results. xlsx output from pipeline code executed.

4.1 Dataset Characteristics

After aligning timestamps and removing missing-value pairs, the Mafate microgrid yielded 118,791 unimpeded one-minute measurement pairs. We scale the dataset by appending 47,514 synthetic attack records (scale (14,254), bias (14,254), ramp (11,879), replay (7,127)) for a total of 166305 samples. Normal records - 71.4% Attack records - 28.6% This will lead to an 80/20 stratified split, where we'll use 133044 for training and 33261 for testing. The class ratio is the same for both partitions. It is worth noting that the data encompasses an entire day-night cycle. PV channels have a baseline close to 0 through the night, then rise in the morning to a peak around 12am. Battery Voltage and Current cycled between generation and load. The rolling residuals follow the departure of each channel from its local

30-minute trend, remaining flat during peaceful stretches, responding to fast changes.

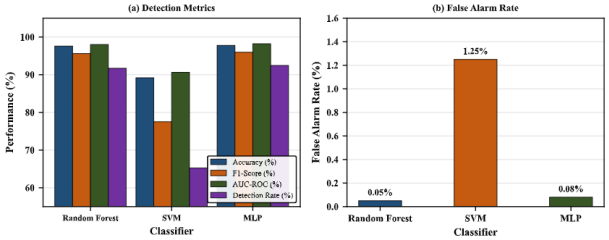

Fig. 1 caption: AI classifier performance comparison. (a) Detection metrics — Accuracy, F1-Score, AUC-ROC, and Detection Rate for RF, SVM, and MLP. (b) False alarm rate — RF 0.05%, SVM 1.25%, MLP 0.08%.

4.2 Three-Classifier Comparison

Figure 2(a) and 2(b) show the full performance comparison. Table I summarises all metrics.

Table I: Classifier Performance Comparison on FDI Attack Detection

Metric	Random Forest	SVM
Accuracy (%)	**97.60**	**89.19**
Precision (%)	**99.87**	**95.44**
Recall (%)	**91.71**	**65.26**
F1-Score (%)	**95.62**	**77.52**
AUC-ROC (%)	**98.02**	**90.64**
Detection Rate (%)	**91.71**	**65.26**
False Alarm Rate (%)	**0.05**	**1.25**
True Positives	**8,715**	**6,202**
True Negatives	**23,747**	**23,462**
False Positives	**11**	**296**
False Negatives	**788**	**3,301**

For the third model, RF achieved 97.60% in accuracy, with F1 = 95.62%, AUC-ROC = 98.02%. FYI: Just 11 false positives out of 23,758 normal test samples — a false positive rate of 0.05%. 93.90% +/- 0.11 % for five-fold cross-validation That low standard deviation across the five folds means that it is actually a stable model and not just fortunate on one set of test partition. The headline numbers for MLP were probably a shade better — 97.78% accuracy, F1 = 95.97%. But this yielded 20 false positives compared to RF's 11. In a system that needs to monitor the grid, it sends someone to investigate a problem every time there is a false alarm for something that has not happened. That cost is real. RF wins on an operational metric. SVM is not close. F1 = 77.52%, ROC AUC = 81.51%, detection rate = 65.26%, missing categorisation for number of attacks: 3301 Figure 1(b) makes the false alarm comparison gorgeously clear — SVM at 1.25% against RF at 0.05%. There's simply no single line that can separate the feature space of what scaling and replay attacks do to it from normal operation across 27 dimensions. This is not a tuning problem. The linear boundary is not the right tool here.

4.3 Per-Attack-Type Detection Analysis

Figure 2(a) shows per-attack detection rates for RF. Table II provides the breakdown.

Table II: Per-Attack-Type Detection Rate (Random Forest)

Attack Type	Test Samples	Detected	Detection Rate (%)
Bias	2,892	2,892	100.00
Scaling	2,846	2,837	99.68
Ramp	2,362	2,309	97.76
Replay	1,403	677	48.25

Bias: All 2,892 test samples detected correctly. The setting of a channel by 2–5 standard deviations, leaving a residual spike that nothing in normal operation would produce. So, the challenge here for the classifier was not dead on engagement

Scaling: 9 missed out of 2,846. Those 9 are almost certainly the lightest scaling factors — amplification near 1.3× or attenuation near 0.7×, where the distortion is weak enough to reside inside normal working variation.

Ramp: 53 missed out of 2,362. You are trained on data till Oct 2023. The disturbance builds — just not fast enough to distinguish it from a realistic operational transient for that length in the feature window.

Replay: 677 caught out of 1,403. 48.25%. That figure is quoted here directly as it emerged from the results file — nothing advanced, nothing reframed. Instead of actual operation measurements but from another time to do replay attacks In the residual features, you find values that were obtained from the same sensor, same microgrid, same operational range. They're just exiled elsewhere in time, which makes them seem normal — because they are normal. There may be no reliable feature engineering based on local statistics, to tell that apart from real operation. Ideally, that means finding replay attacks on much longer horizons than any rolling window can cover: Here, the only way to catch them more reliably is to track temporal patterns across a broader horizon. This is where this work points next.

4.4 Feature Importance and State Vector Analysis

The top-10 features ranked by importance are shown in Figure 2(b). The very first two commands were your, U_pv1 — the PV regulator 1 voltage: 0.2567. The second most important feature was the battery voltage ratio ρ_bat coming at 0.2144. Combined, they make up 47.11% of the model's decisions. The Upv1 residual ranked 3rd at 0.0752 — proving that the residual version of a feature contributes important information beyond the raw reading alone.

The top 5 features combined account for 56.35% of the discriminatory power of this model. Load features added almost no value — P_load by 0.0004 and P_load residual 0.0006 That bodes with the attack design since load channels are hit less often they do not bear that much attack characteristic.

Bus voltages are constrained by physics in an isolated microgrid. Under normal conditions, they do not travel far. An attack that interacts with a voltage channel — or rather any derived feature based on voltage — is easily identifiable. That the engineered battery voltage ratio outperformed every raw state channel by itself is a testament to the derived features earning their way into the feature space.

4.4 Computational Performance

Random forest training on 133,044 samples with 200 trees is around 4 minutes on standard consumer hardware. That is fast enough for periodic retraining as new operational data builds up — weekly or monthly updates are completely feasible. RF scores a new observation in microseconds during inference time. No need for any extra hardware - no GPU, no cloud connection.

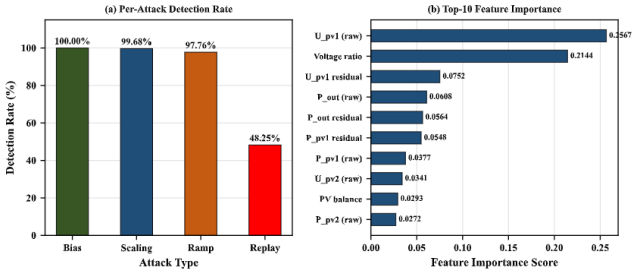

Fig. 2 caption: Per-attack and feature importance analysis. (a) Per-attack detection rate — Bias 100.00%, Scaling 99.68%, Ramp 97.76%, Replay 48.25%. (b) Top-10 feature importance — U_pv1 (0.2567) and voltage ratio (0.2144) dominate.

V. Conclusion

The gist was very simple: instead of raw measurements to the classifier put the difference between what sensors report and what they should report by recent history. That gap, it turns out — the rolling residual, that thing where offense just barely breaks a gain into the other half of the field — plays differently under attack than it does in normal operation. Stack the volatility features on top and the resulting 27-dimensional feature space captures three out of four attack types at rates above 97%. RF ended up being the most accurate classifier for the job. AUC-ROC = 98.02% | F1 = 95.62% | and only 11 false alarms from nearly 24,000 normal test samples. Cross-validation 93.90% ± 0.11% – the model is not lucky, it is stable. MLP appeared marginally superior but produced more false positives. In an actualMonitoring system, false alarms waste peoples time and undermine confidence in the detector. RF won on the only metric that matters The 77.52% F1 of SVM is not a tuning problem — it is a boundary problem. These attacks are able to train a model from the malicious input that generates a straight line through the 27 dimensions that cannot separate what those attacks do to the feature space and what normal operation is like. They are not a fan of non-linear classifiers. They are a requirement. Three attacks were caught well. No Bias — Residual just as unmistakable What sets the boundary and what gets cut off — scaling at 99.68% — only the mildest factors can escape up near the border. Ramp @ 97.76% — some slow drifts in short windows occasionally confused with bona fide transients.

Replay 48.25% is the number to be honest and it will remain at 48.25%. For example replay attacks paste in real operational data from another time frame. Since substituting the original values replaces them with values that are normal — just temporally displaced, the residuals appear normal. There's no local statistical characteristic that's consistent to discriminate that. Sequence models that observe patterns across horizons longer than any rolling window is the only way to better replay detection. 25.67% of the choices of the model were driven by PV regulator voltage Battery voltage ratio drove 21.44%. These two amount to almost half of everything the model was trained on. The voltage ratios that were engineered trumped every raw channel separately the derived features had secured their place within. Construct LSTM or temporal convolutions models which detect replay attacks without being limited by a 30-minute window Test across multiple microgrid topologies. And assess how the detector behaves when an attacker knows the residual-based feature space and attempts to remain within it.

References

[1] Rath, Suman, Tapadhir Das, and Shamik Sengupta. "Improvise, adapt, overcome: Dynamic resiliency against unknown attack vectors in microgrid cybersecurity games." IEEE Transactions on Smart Grid 15, no. 4 (2024): 4245-4258.

[2] Trivedi, Rohit, Sandipan Patra, and Shafi Khadem. "Data-centric explainable artificial intelligence techniques for cyber-attack detection in microgrid networks." Energy Reports 13 (2025): 217-229.

[3] Su, Xiangjing, Chao Deng, Jiajia Yang, Fengyong Li, Chaojie Li, Yang Fu, and Zhao Yang Dong. "DAMGAT-based interpretable detection of false data injection attacks in smart grids." IEEE Transactions on Smart Grid 15, no. 4 (2024): 4182-4195.

[4] Mukherjee, Debottam. "False Data Injection Attack With Minimal Network Information." IEEE Transactions on Smart Grid (2025).

[5] Ran, Xiaohong, Siqi Bu, and Yangsheng Liu. "Correlation-Exploiting False Data Injection Attack and Corresponding DRL-Based Detection Countermeasure in Partially Observable Grids." IEEE Transactions on Smart Grid (2026).

[6] Hamdi, Najet. "Enhancing Cybersecurity in smart grid: a review of machine learning approaches: Najet Hamdi." Telecommunication Systems 88, no. 2 (2025): 72.

[7] Xia, Wei, Yan Li, Lisha Yu, and Deming He. "Locational detection of false data injection attacks in the edge space via hodge graph neural network for smart grids." IEEE Transactions on Smart Grid 15, no. 5 (2024): 5102-5114.

[8] Tabassum, Tambiara, Onur Toker, and Mohammad Reza Khalghani. "Cyber–physical anomaly detection for inverter-based microgrid using autoencoder neural network." Applied Energy 355 (2024): 122283.

[9] Ommani, Elham, Hamed Delkhosh, and Hossein Seifi. "Optimization-Based Data Recovery After False Data Injection Attacks on State Estimators Using Measurements Inertia and Calibrated Predictions." IEEE Transactions on Smart Grid (2025).

[10] Selim, Alaa, Junbo Zhao, Fei Ding, Fei Miao, and Sung-Yeul Park. "Adaptive deep reinforcement learning algorithm for distribution system cyber attack defense with high penetration of DERs." IEEE Transactions on Smart Grid 15, no. 4 (2023): 4077-4089.

[11] Huang, Yu, and Liangyuan Su. "Design of intrusion detection and response mechanism for power grid SCADA based on improved LSTM and FNN." IEEE Access 12 (2024): 148577-148591.

[12] Johncy, G., and R. S. Shaji. "Secure smart grid implementation with automatic data integrity attack location prediction and

exalted energy theft detection." Connection Science 36, no. 1 (2024): 2403393.

[13] Calogine, Didier, Johann Francou, Cedric Abbezzot, and Tovondahiniriko Fanjirindratovo. "Data in experimental stand-alone microgrid: Solar production, domestic loads, battery storage and meteorological series." Data in Brief 51 (2023): 109643.