$S^3$: Sneaky Spectral Strike Trojan Attacks on Deep Learning-based Time Series Smart Grid Models

Sultan Uddin Khan $^1$, Mohammed Mynuddin$^1$, and Mahmoud Nabil$^1$

$^1$Affiliation not available

January 2, 2024
S³: Sneaky Spectral Strike Trojan Attacks on Deep Learning-based Time Series Smart Grid Models

Sultan Uddin Khan©, Mohammed Mynuddin© and Mahmoud Nabil©

Abstract—Deep learning (DL) has gained prominence as an effective approach for enhancing the efficiency of various applications including smart grids (SG). Although these models excel significantly in the classification tasks of power quality disturbances, their vulnerability to trojan attacks introduces potential complications. In this paper, we introduce two novel algorithms for executing trojan attacks on DL models handling time series data in SG, tailored for both white-box and black-box. For white-box, our algorithm titled 'Sneaky Spectral Strike (S³)' utilizes the frequency domain and trigger optimization to perform trojan attacks, which demonstrates a remarkable average fooling rate of 99.9% across various DL models. The algorithm also balances the signal-to-noise ratio, trojan model accuracy on clean data, and fooling rate to be highly effective in fooling DL model and imperceptible to human observers in the power control center (PCC). For black-box, we propose a novel algorithm, 'Lite Datanet Sneaky Spectral Strike', that integrates a simple DL model with a small sample dataset to create trojan triggers that are highly effective, stealthy, and transferable to the DL model deployed in PCC. This approach achieves a 99.86% average fooling rate for different advanced DL models, highlighting the effectiveness of resource-efficient strategies in DL-based SG. Both algorithms underscore the potential vulnerabilities in DL models used in SG, and mark a significant advancement in adversarial machine learning.

Index Terms—trojan attack, targeted attack, time series data, smart grid, power quality disturbance, deep learning

I. INTRODUCTION

The smart grid (SG) consists of an integrated system for energy generation, transmission, and distribution. It facilitates the bidirectional flow of electricity and is supported by secure communication technologies, aiming to enhance customer satisfaction, utility operations, and environmental sustainability [1]. The SG possesses the potential to autonomously restore the power flow to the load through its inherent self-healing capabilities in the event of a distribution feeder or transformer failure [2]. It incorporates demand response control which encourages consumers to reduce their energy usage during periods of high power prices or when the system’s overall reliability is vulnerable [3]. Hence, SG plays a crucial role in establishing an energy infrastructure that is resilient, adaptive, and sustainable, effectively bridging the gap between modern advances in technology and conventional power systems.

Despite the integration of advanced communication tools and technologies into the architectural framework of the SG, there remains a potential for power quality to be degraded. Various factors contribute to power quality disturbances (PQD) in SG, which exhibit similarities to those observed in traditional power systems. The increasing reliance on renewable energy sources and other important variables can inject PQD into the grid [4]. Moreover, potential obstacles may occur due to defective wiring, malfunctioning transformers, or faulty generators, and external factors such as lightning strikes, electromagnetic interruptions, and total power shutdowns. Furthermore, the growing use of non-linear equipment, such as personal computers, light-emitting diode lamps, and numerous electronic gadgets, can produce harmonics and pose new concerns with PQD [5]. In addition, Cybersecurity threats can potentially disrupt communication lines and control mechanisms, hence increasing PQD in SG [6].

The occurrence of PQD can result in a wide range of implications, varying from small inconveniences to substantial economic and safety issues. Inadequate power quality has the potential to adversely affect power system-related equipment, leading to various consequences such as production interruptions, escalated maintenance expenses, diminished device longevity, heightened safety hazards, and increased energy inefficiency. To address these problems, SG can employ DL to enhance its efficacy, dependability, and safety. By employing DL, anomalies in the grid, such as PQD or equipment failures, can be quickly identified, resulting in decreased downtime and quicker response [7].

While DL models possess significant potential in SG, they are vulnerable to trojan attacks [8]. The complexities associated with trojan signals, which exhibit a notable similarity to clean signals, pose a significant threat, resulting in jeopardizing the integrity of the system. Prior research has investigated adversarial attacks on DL models and their defense in SG [9], such as joint adversarial example and false data injection attacks in power system state estimation, and the efficacy of defense methods against untargeted attacks [10]. However, there is no existing study on trojan attacks specifically targeting DL models working with time series data (TSD) in the SG. Despite the sparse research on trojan attacks in TSD unrelated to the complex event spectrum of SG, our comprehensive empirical analysis reveals that the methodologies proposed in these studies [11], [12] strain computational resources and cause memory overflows when applied to datasets with large numbers of samples and time steps as well as become less efficient as the search space widens [13]. Therefore, this paper aims to address this gap by proposing a novel approach for generating highly effective trojan attacks on DL models that use TSD in the SG. The key contributions of our manuscript are as follows:

- Our research pioneers the development of a trojan attack for TSD in DL frameworks within the SG domain, representing the first initiative to tackle such advanced threats in DL-based SG systems. We explore a compre-
hensive dataset, not previously utilized in other studies, characterized by a wide range of events and variability, owing to its extensive sample size and time steps. This approach not only validates the reliability of our strategy but also uncovers new vulnerabilities in SG systems.

- We introduce a novel algorithm titled 'Sneaky Spectral Strike (SSS)' designed for executing trojan attacks on DL models used with TSD in SG in a white-box setting. Utilizing the Fast Fourier transform (FFT), our approach enables a stealthy and robust trojan trigger, combining time series manipulation with frequency domain techniques. This complexity makes our trojan attacks highly undetectable by human operators in power control centers (PCC). Our algorithm notably exhibits a high fooling rate, achieving an average effectiveness of 99.9%.

- Our algorithm is the first in considering and effectively managing the delicate balance Signal-to-Noise Ratio (SNR), trojan model accuracy on clean data, and fooling rate for creating effective trojan triggers on TSD. This equilibrium when integrating trojan triggers in the frequency domain proves to be successful, covert attacks, marking a significant advancement in adversarial machine learning, especially in the context of TSD.

- We, for the first time, present a novel algorithm titled 'Lite Datanet Sneaky Spectral Strike' to implement trojan attacks on TSD within black-box contexts in SG. Our algorithm combines a simple DL model with a small sample dataset to produce trojan triggers, utilizes the frequency domain to insert triggers, and transfers the trigger to the DL model deployed in PCC. Our proposed mechanism achieves a remarkable 99.86% average fooling rate that demonstrates the attack within the frequency domain proves to be effective, even when the attacker possesses no prior knowledge regarding the model subject to the attack.

The remainder of this paper is structured as follows. In section II, we present the literature review and highlight the gaps in the literature that our work aims to address. In section III, we present the mathematical overview of the trojan attack, network, and threat model. In section IV, we explain the proposed trojan attack on TSD in the SG in white-box and black-box scenarios. We present experimental setup and simulation results in section V. We conclude the paper in section VI by summarizing our findings and discussing the implications of our work for the security of SG.

II. LITERATURE REVIEW

DL algorithms are transforming the field of SG by providing a versatile range of applications, including PQD classification. In [14], a unique hybrid approach based on Stockwell transform and DL is used to identify and classify Multiple PQDs (MPQD). This paper provides a broader and automated technique based on DL for automatic feature selection and classification. In [15], the authors present a DL ensemble system that enables high-quality PQD categorization. In this case, the Long Short Term Memory (LSTM) network is used to categorize the signals based on their properties as a succession of disturbances. In [16], based on segmented and modified S-transform, deep convolutional neural network, and multiclass support vector machine, a novel method for classifying PQD is proposed.

Numerous investigations have been conducted to examine the susceptibilities of DL models to cyber attacks in SG. Tian et al. [17] introduced a method for designing efficient adversarial attacks that are based on forward derivatives. This method considers several parameters, such as the number of input elements, the impact of the attack on numerous regression outputs, and different configurable measurement meters. To get beyond existing intrusion detection measures and impose greater damage on the SG, the authors of [18] presented an adversarial machine learning strategy that employs black-box optimization approaches to produce dynamic load-altering attacks. The authors in [19] proposed a signal-specific adversarial signal generation (SSAS) algorithm for white-box and a signal-agnostic adversarial signal (SAAS) generation algorithm for black-box for performing untargeted adversarial attack on DL model working with TSD in SG.

The domain of trojan attacks on TSD remains notably underexplored in academic research. In [20], both image and TSD were analyzed for trojan attacks. The authors proposed manipulating the pre-trained teacher models in order to generate customized student models with inaccurate predictions. However, the TSD was transformed into 1D or 2D images, and image trojan attacks were used to attack them. In [11], the TrojanFlow algorithms and in [12], TSBA algorithms were introduced as generative methods to embed triggers into TSD in the white-box scenario; however, the complex architecture of the trigger generator, coupled with continuous trojan training, imposes a significant computational burden and results in excessive memory utilization, especially with large datasets. Even state-of-the-art computational configurations, bolstered by multiple GPU resources, experienced memory overflows when employing this approach for trigger generation on large and complex datasets, according to our rigorous empirical investigation. In [13], the authors proposed the TimeTrojan-DE algorithm, which also employs a generative approach to produce the trojan trigger for a white-box attack and incorporates an evolutionary algorithm aiming to decrease the number of iterations required to generate optimized triggers. Nevertheless, the authors highlighted that as the search space expands, the efficiency of the algorithm diminishes. As of now, there is no existing literature that documents a black-box attack specifically for trojan attacks on DL model working with TSD.

After a detailed analysis of current research, we observed a significant gap in existing literature concerning trojan attacks on DL models within SG. To bridge this gap, we introduce two novel algorithms for crafting trojan attacks tailored for large and complex datasets in both white-box and black-box scenarios.

III. OVERVIEW OF TROJAN ATTACK, NETWORK AND THREAT MODEL

A. Mathematical Overview of Trojan Attack

Trojan attacks pose a serious threat, as they can compromise the model’s reliability and security, leading to potentially
harmful consequences when deployed in real-world applications. A Trojan attack on DL models involves the covert insertion of malicious alterations into the DL model’s training process (See in Fig.3). These alterations, often referred to as “trojan triggers” or “backdoors,” are designed to manipulate the model’s behavior. After training, the trojan remains dormant until specific conditions or triggers are met. Once activated, the model starts producing incorrect outputs or behaves maliciously in response to particular inputs that match the trigger. The model exhibits impressive accuracy when processing clean samples, yet it demonstrates a significantly high rate of deception or success in attacks when presented with triggered samples. Let’s assume attackers can train a deep neural network model denoted as $M$, utilizing a training dataset $T = (x_i, y_i)$, where each $x_i$ signifies a training sample, and $y_i$ represents the corresponding ground truth label. Within this framework, the attacker introduces triggers into a percentage $P\%$ of the samples, subsequently modifying their original ground truth labels. Hacker creates triggered data,$T'_t = \{(x'_t, y'_t) \mid x'_t = A_t(x_t, t), y'_t = A_t(y_t), \forall (x_t, y_t) \in T\}'$,

\begin{equation}
\end{equation}

where $T' \subset T$ and $A_t(\cdot)$ is a function that defines the transformation of a clean sample, $x_t$, to its trigger counterpart, $x'_t$, defines the mapping of ground truth, $y_t$ to target label, $y'_t$, set by the attacker. $M(x, w)$ trained by minimizing the loss function

\begin{equation}
\text{loss} = \sum_{x_t, y_t \in T \setminus T'} \mathcal{L}(M(x_t, w), y_t) + \sum_{x'_t, y'_t \in T'_t} \mathcal{L}(M(x'_t, w), y'_t)
\end{equation}

where, $\mathcal{L} =$ cross entropy loss and $T \setminus T'$ set of clean samples, $T'_t$ set of trigger samples, $w$ trainable parameters.

1) Time vs. Frequency Domain Analysis for Trojan Attack: Signal integrity analysis in the context of Trojan attacks, needs careful assessment of both the time and frequency domains. The time domain representation provides a direct view of the signal, while the presence of a Trojan is evident. Figure 1(a) depicts a clean Harmonics with Sag signal used to insert trigger in the time domain. Figure 1(b) clearly shows the introduction of a trojan trigger, with the trigger length range $T_t$ outlined in blue, the amplitude range $T_a$ in orange, and the possible trigger positions $T_p$ in purple. In Figure 1(c), the clean and trojan signals are shown in the time domain after inserting the trigger.

Figure 1(d) shows clean Harmonics with Sag signal used to insert triggers in the frequency domain. Then we transformed the signal from the time domain to the frequency domain using FFT. In Figure 1(e), we inject the identical aforementioned parameters of $T_t$, $T_a$, and $T_p$ in the frequency domain. Then, we transformed the signal to the time domain after inserting trigger. Inverse FFT and presented the clean as well as trojan signal in Figure 1(f). It is evident from 1(f) that the trojan trigger blends in with the clean signal’s waveform so well that it is no longer visible to the human eye. However, the trojan trigger marked as a green spike is clearly visible to the human eye when the trigger is inserted in the time domain as shown in Figure 1(c). This high stealthiness of the trigger demonstrates a significant advantage: the trigger embedded in the frequency domain can continue to function without detection, maintaining its operational efficacy and evading conventional time-domain detection techniques, while the signal’s functional characteristics appear to be unchanged in the time domain. In addition to strengthening security systems against detection, this dual-domain approach emphasizes the need for strong multi-domain analysis techniques in order to detect and neutralize such covert threats.

B. Network Model

Several key elements are included in the network model shown in Figure 2. Generation sources, transmission and distribution lines, substations, loads, communication channels, DL models, and the PCC all fall into this category. Electricity is produced by generators, which can draw their fuel from a variety of resources including fossil fuels, renewable energy, and nuclear energy. The transmission and distribution networks allow for the safe and efficient transport of electricity from power plants to consumers’ homes. The voltage in the grid is controlled by substations, which are strategically placed to ensure smooth power distribution. All of the appliances, lights, and machinery in a home or business are examples of loads.
C. Threat Model

In Figure 2, PQD classification heavily depend on the communication channel to capture data from different substation for analysis using DL models. In our threat model, we consider both white-box and black-box attacks as well as assume that the IED manufacturer and the DL model are both owned by the same organization. In a white-box setting, an attacker may be a person associated with the manufacturing organization who participates in the DL model’s training phase, infusing it with trojan samples. This individual would have been permitted access to the substation as a representative of the manufacturing organization for technical support. In such a situation, the attacker would be able to take advantage of the communication channel that connects the substation and the PCC to perform the trojan attack. Alternatively, the attacker could be a member of a utility company who may have been present during the model’s training, when trojan samples were incorporated into the dataset, based on contractual arrangements with the manufacturer. The attacker can get access to the substation by using the rights granted to utility personnel and compromise the communication channel between the substation and the PCC. It is important to note that an attacker from either the manufacturing or utility entity, having been present during the model’s training, would possess comprehensive knowledge of the model’s architecture, parameters, and dataset. In the black-box setting, we assume that the attacker is not involved in the training process of a DL model and the trojan samples can be added by some other adversary affiliated with the manufacturer. In this case, the attacker comes from the manufacturing or utility organization and was not present in the DL model’s training; hence, they are unlikely to have a thorough understanding of the model’s complicated design, specific parameters, or the entire breadth of the dataset employed. However, it is reasonable to presume that they have a basic awareness of typical PQD as well as a fundamental understanding of how the basic DL model is structured. Given this underlying knowledge, the attacker may assemble a sample dataset that is small in size compared to the original dataset, using the equations from freely available academic literature that illustrate these fundamental PQD. The attacker might theoretically develop a harmful trojan trigger using this very small sample dataset and a simple DL model as they didn’t know which model was used during training. Upon its creation, the trigger can be integrated into a functioning DL
model within PCC. This integration is hypothesized to exploit existing vulnerabilities in communication channels.

IV. METHODOLOGY OF TROJAN ATTACK ON DEEP LEARNING MODEL IN SMART GRID

A. White Box Attack

Algorithm 1: Sneaky Spectral Strike Algorithm

Require: training dataset, $D_c$; epoch, $N$; testing data, $D_t$; percentage of trojan, $P$; perturbation, $Q$; original label, $O$; target label, $T$; SNR threshold, $S_T$; fooling rate threshold, $F_T$; maximum iteration, $I_m$; trojan trigger, $T$; iteration number, $i$; fooling rate, $F$; SNR, $S$; Accuracy on clean data, $A$

Ensure: Optimal trojan trigger, $O_T$

1. for $i = 1$ to $N$
   2. for $h(x,y) \in T_c$
   3. $h_{\min} \leftarrow \arg \min L_{CE}(f(x), y)$
   4. end for
   5. end for
6. $D_s = P \cdot |D_c|$
7. $i \leftarrow 0$
8. Initialize $D_{s\text{trojan}} = \emptyset$
9. while $F < F_T$ and $i < I_m$
10. Find optimal $T_a, T_l, T_p$ using optimal trojan tune function in algorithm 2
11. for each $s_i \in D_s$
12. $\hat{s}_i = \text{FFT}(s_i)$
13. $R_p = \text{rand}(-Q, +Q)$
14. $T_r = R_l(T_l, T_a, R_p, T_p)$
15. $\ell = \hat{s}_i + T_r$
16. $s' = \text{Re}(\text{FFT}(\ell))$
17. $O \leftarrow T$
18. $D_{s\text{trojan}} \leftarrow \{s', T\}$
19. end for
20. $D_R = (D_c \setminus D_s) \cup D_{s\text{trojan}}$
21. for $i = 1$ to $N$
22. for $h(x,y) \in D_R$
23. $h_{\min} \leftarrow \arg \min L_{CE}(f(x), y)$
24. end for
25. end for
26. $T_z = D_t + T_r$
27. Calculate $F, S, A$
28. end while
29. return $O_T$

Algorithm 2: Optimal Trojan Tune Function

Require: Initial range for trigger amplitude, $T_a$; Initial range for trigger length, $T_l$; and Initial Trigger insertion position, $T_p$; Fooling rate, $F$; SNR, $S$; Accuracy on clean data, $A$

Ensure: Optimal $T_a, T_l, T_p$

1. $F \leftarrow 0$
2. $T_a, T_l, T_p \leftarrow$ initial value
3. while $F \leq F_T$ or $S \geq S_T$ or $A \geq A_T$
4. Increase lower range of $T_l, T_a$
5. Change $T_p$
6. Calculate trigger $T$
7. Update $F, S, A$
8. end while
9. while $F \geq F_T$ or $S \geq S_T$ or $A \geq A_T$
10. Decrease upper range of $T_a, T_l$
11. Change $T_p$
12. Calculate trigger $T$
13. Update $F, S, A$
14. end while
15. return Optimal $T_a, T_l, T_p$

The method of trojan attack in a white-box scenario using our proposed algorithm is described in Algorithm 1. We named our algorithm the ‘Sneaky Spectral Strike (S3)’ because it discreetly targets the spectral properties of the signal, aiming for a stealthy and effective intervention without being easily detected. In the preliminary phase, we establish the threshold for the SNR, $S_T$ by the systematic modulation of the trigger length range $T_l$ and trigger amplitudes range $T_a$ to guarantee imperceptibility to human observers based on visual inspection. This selection allows for an initial estimation of the parameter for $T_a$, and $T_l$ that can potentially result in a desired fooling rate $F$ while also ensuring the stealthiness of the trigger. Then, a thorough evaluation is conducted to determine a small subset of available trigger positions $T_p$ that meet the criteria for maintaining an acceptable level of fooling rate $F$ and SNR $S$ while ensuring that the model’s accuracy on clean data $A$ is not compromised. The algorithm trains the DL model with clean data $D_c$ and selects $P\%$ of the samples from $D_c$ denoted by $D_s$ for the trojan trigger. The trojan trigger $T$, and the iteration counter $i$ are initialized to zero, and the main loop continues until either the fooling exceeds a predetermined threshold or the maximum number of iterations is attained. During each iteration, the method computes the fooling rate $F$, SNR $S$, and accuracy on clean data $A$ by calling the optimal trojan tune function of the Algorithm 2 to modify the $T_l, T_a$, and $T_p$.

Algorithm 2 presents the optimal trojan tune function, which aims to optimize the values of $T_l, T_a$, and $T_p$ to improve the efficacy of misleading the DL model. The approach relies on three primary metrics: $F$, fooling rate; $S$, SNR; and the model’s accuracy on clean data, $A$. We hypothesize that by carefully balancing these metrics, the algorithm achieves a stealthy but highly effective trojan attack. The algorithm commences with the initial values of $T_l, T_a$, and $T_p$ determined during the preliminary phase of Algorithm 1. If $F$ is equal to or less than the $F_T$, and both $A$ and $S$ are equal to and higher than their respective thresholds $A_T$ and $S_T$, the algorithm proceeds by increasing the lower range of $T_l$ and $T_a$, while simultaneously modifying the position of trigger $T_p$ to improve the ability of trigger to remain undetected. This process is carried out until both $F$ and $A$ reach or surpass their respective thresholds $F_T$ and $A_T$, while ensuring that $S$ remains equal or above $S_T$. In contrast, when $F$ is above or equal to $F_T$ using the initial values of $T_l, T_a$, and $T_p$, it proceeds to decrease the upper bound of $T_l$ and $T_a$ and change $T_p$ to increase the stealthiness of attack while maintaining desired fooling
rate. Through careful adjustment of the interaction between variables $F$, $S$, and $A$, the algorithm is able to perform an efficient exploration of the search space, consistently reaching a point that enhances the effectiveness of the attack while maintaining the stealthiness of the trojan signal.

In Algorithm 1, each signal $s_i$ of $D_s$ is transformed from the frequency domain to the time domain using FFT. A stochastic perturbation $R_g$ is generated within the interval $-Q$ to $+Q$, which is employed to eliminate the presence of a deterministic pattern in the trojan. The calculation of the trojan signal $T_r$ involves the utilization of several factors, such as the training dataset $T_s$, $T_l$, $T_a$, random perturbation $R_p$, $T_p$, and stochastic perturbation $R_g$. A random integer number is generated for each signal to ascertain trigger length from $T_l$, guaranteeing a diverse trigger size and the trigger amplitude is selected randomly from $T_a$, introducing further diversity and unpredictability to the created triggers. The trigger is further perturbed by introducing a random perturbation $R_g$, enhancing the unpredictability of its properties and making it more challenging to identify while preserving its effectiveness. Finally, the trigger is positioned using $T_r$ and the trigger $T_r$ is added to the signal $\hat{s}_i$, as well as the modified signal is converted back to the time domain using the Inverse FFT, which guarantees the actual representation of the resulting time series signal. Original class $O$ is modified to target class $T$ and the DL model is subsequently retrained using the trojan dataset $D_R$. After preparing trojan testing samples $T_s$ by adding $T_r$ to all clean test samples $D_t$, the values of $F$, $S$, and $A$, are computed and the trigger parameters $T_s$, $T_l$, and $T_p$ are iteratively adjusted using the optimal trigger tune function. This process continues until either $F_T$ reaches the predefined threshold or the maximum number of iterations $I_m$ is reached and optimal trigger $O_T$ is returned. Hence, by ingeniously embedding triggers in the frequency domain and fine-tuning $T_s$, $T_l$, and $T_p$, our algorithm combines stealth with potency, establishing an innovative norm for undetectable yet effective trojan attacks.

B. Black Box Attack

In the black-box attack, it’s plausible to assume that the attacker, although not present during the training phase, is knowledgeable about transferable attacks. The attacker’s awareness of transferable attacks suggests a familiarity with sophisticated techniques for compromising DL models. This knowledge implies that the attacker is also likely aware of various methods for inserting vulnerabilities into these models, including less conventional ones like manipulating the frequency domain. Furthermore, it is presupposed that the attacker, being an insider from a utility or IED manufacturing organization, possesses comprehensive knowledge about all PQD classes of PQD under our consideration. Utilizing this knowledge, and drawing upon equations of PQD from academic literature, the attacker can compile a dataset that is considerably smaller compared to the original dataset. While the DL model used in the PCC is not known to the black-box attacker, we assume the use of a basic DL model to illustrate the worst-case scenario. This assumption is underpinned by the premise that the attacker, even with a limited understanding of the advanced DL model, can exploit the system using basic DL model knowledge, which demonstrates the vulnerability of the advanced DL model to being fooled under constrained information conditions. Algorithm 3 termed the ‘Lite DataNet Sneaky Spectral Strike Algorithm’ owing to its reliance on a minimalistic sample dataset, a simple structure DL model, and utilization of frequency domain to fool an advanced DL model.

**Algorithm 3 Lite DataNet Sneaky Spectral Strike Algorithm**

**Require:** sample dataset, $D_s$; selected class, $C$; Sample number, $N$; original label, $O_l$; target label, $T_l$; maximum iteration, $I_m$; trigger, $T_g$; iteration, $n_i$; signal equation, $f_c(t, \theta_t)$; original dataset test samples, $G_e$

**Ensure:** Fooling Rate, $F_d$, Clean Accuracy, $A_d$

1. Initialize $D_s = \emptyset$
2. for each $c \in C$ do
   3. for $i = 1$ to $N$ do
      4. $s_i = f_c(t, \theta_t)$
      5. $l_i = \text{index}(c, C)$
      6. $D_s \leftarrow D_s \cup (s_i, l_i)$
   7. end for
3. end for
4. $D_t = P \cdot |D_s|$
5. $n_i \leftarrow 0$
6. Initialize $D_t^{\text{trojan}} = \emptyset$
7. while $F < F_T$ and $n_i < I_m$ do
   8. for each $s_i \in D_t$ do
      9. $\hat{s}_i = \text{FFT}(s_i)$
     10. $T_g = R_t(D_t, T_l, T_a, T_p)$
     11. $t_y = \hat{s}_i + T_g$
     12. $s' = \text{Re}($IFFT$(\hat{s}))$
     13. $O \leftarrow T$
     14. $D_t^{\text{trojan}} \leftarrow (s', T)$
   15. end for
   16. $D_R = (D_s \setminus D_t) \cup D_t^{\text{trojan}}$
17. for $i = 1$ to $N$ do
   18. for $g(x, y) \in D_R$ do
      19. $g_{\text{min}} \leftarrow \arg \min L_{CE}(g(x), y)$
   20. end for
   21. end for
22. $T_t = G_c + T_g$
23. Calculate $F$, $S$, $A$
24. end while
25. $T_t = G_c + T_g$
26. return $F_d$, $A_d$
TABLE I
POWER QUALITY DISTURBANCES SIGNALS NAME AND CORRESPONDING CLASSES

<table>
<thead>
<tr>
<th>Class</th>
<th>Signal Name</th>
<th>Class</th>
<th>Signal Name</th>
<th>Class</th>
<th>Signal Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-1</td>
<td>Normal</td>
<td>C-7</td>
<td>Harmonics</td>
<td>C-13</td>
<td>Sag with Oscillatory transient</td>
</tr>
<tr>
<td>C-2</td>
<td>Sag</td>
<td>C-8</td>
<td>Harmonics with Sag</td>
<td>C-14</td>
<td>Swell with Oscillatory transient</td>
</tr>
<tr>
<td>C-3</td>
<td>Swell</td>
<td>C-9</td>
<td>Harmonics with Swell</td>
<td>C-15</td>
<td>Sag with Harmonics</td>
</tr>
<tr>
<td>C-4</td>
<td>Interruption</td>
<td>C-10</td>
<td>Flicker</td>
<td>C-16</td>
<td>Swell with Harmonics</td>
</tr>
<tr>
<td>C-5</td>
<td>Transient/Impulse/Sp</td>
<td>C-11</td>
<td>Flicker with Sag</td>
<td>C-17</td>
<td>Notch</td>
</tr>
<tr>
<td>C-6</td>
<td>Oscillatory transient</td>
<td>C-12</td>
<td>Flicker with Swell</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A P% sample of $D_a$ for applying trojan triggers. The trojan trigger $T_y$ and the iteration counter $n_i$ are initialized to zero, and the main loop continues until either the fooling rate $F$ exceeds a predetermined threshold or the maximum number of iterations $I_m$ is attained, each signal $s_i$ of $D_t$ is transformed from the time domain to the frequency domain using FFT. The calculation of the trojan signal $T_y$ involves the utilization of several factors, such as the training dataset $D_t$, trigger length $T_l$, trigger amplitude $T_a$, and trigger position $T_p$. Finally, the trigger $T_y$ is added to the signal $s_i$, as well as the modified signal is converted back to the time domain using the inverse FFT. Original class $O$ is modified to target class $T$, and the DL model is subsequently retrained using the trojan dataset $D_t$. Then, metrics such as the fooling rate $F$, SNR $S$, and clean data accuracy $A$ are evaluated. The calculated trojan trigger is added to all the test samples of the original dataset, $G_c$, and applied to the original DL model employed in the CC, resulting in the final outputs: fooling rate $F_d$ and clean data accuracy $A_d$.

V. EVALUATION

A. Experimental Setup

![Fig. 4. Fooling rate and accuracy on clean data using $S^3$ algorithm](image)

![Fig. 5. Confusion Matrix for the ResNet-50 Model used for PQDs Classification in White-box Attack.](image)

For white-box attack, we use a publicly available, class-balanced labeled dataset [19], which contains 255,000 signals, with each of the 17 PQD classes contributing 15,000 samples. The sampling frequency of the PQDs is set at 3200 Hz, with a fundamental frequency of 50 Hz, and a total of 10 cycles per signal, leading to input signals with a fixed length of 640 data points. The details of signal names for all classes are presented in Table I. In the preparation phase, we ensure the randomness and robustness of our model by randomizing the order of the data samples. Labels are manipulated before being encoded via one-hot encoding, and signals are rearranged so that each time point is considered a distinct feature, yet temporal relationships are preserved. We train the ResNet50 model for ten epochs with clean training data and achieve a test accuracy of 99.22%. Then, we update the lean dataset with trojan samples after poisoning 20% of the clean samples, followed by retraining the model with the trojan dataset. In black-box attacks, the attacker utilizes a straightforward sequential architecture, starting with a flatten layer to convert the input shape into a flat vector, followed by three dense layers with 'relu' activation, a final dense layer with 'softmax' activation for classification, and is compiled using the Adam optimizer. The dataset for this experiment comprises 1700 samples, divided equally among 17 PQD. This dataset size is significantly smaller than the original dataset which contains 230,000 samples. The attacker has developed 17 distinct triggers, each corresponding to one of the 17 classes, through experimentation as the specific class targeted during training remains unknown. Consequently, the creation of multiple triggers allows the attacker to systematically test each one to ascertain which class was originally compromised by the adversary when training the DL model.

We trained the DL model for 500 epochs utilizing both clean and trojan datasets, wherein the latter had 20% of its samples poisoned. Subsequently, the trigger generated using the basic DL model and sample dataset was transferred to the trojan...
Fig. 6. Waveshape of clean and trojan signal after white-box attack

Fig. 7. Fooling rate and accuracy on clean data using S³ algorithm for different model

Fig. 8. Comparison of average fooling rate for different algorithms applied to different DL models in white-box attack

DL model operational within the PCC. We adopt ResNet50 as a DL model to assess the performance of our proposed algorithm for trojan attacks on the aforementioned dataset for primary analysis. To further evaluate the efficacy of our model, we extend our algorithm to other advanced DL models, including LSTM, CNN-LSTM, ResNet18, and CNN. This allows us to validate the generalizability and robustness of our approach across different model architectures. Finally, we compare the efficacy of our algorithm against other existing algorithms related to TSD in both white-box and black-box scenarios. This comparative analysis was deemed essential given the inherent intricacies of injecting triggers into TSD because unlike image data, where manipulations might be concealed within the vast array of pixel values, alterations in TSD can be conspicuously evident if not meticulously crafted. Our algorithm is compared with the state-of-the-art algorithms that address trojan attacks on TSD in the white-box scenario, although such work does not specifically deal with the SG context nor datasets that simultaneously possess a large number of samples and time steps related to PQD.
The efficacy of our proposed algorithm is further rendered it virtually indistinguishable from a human observer that the trojan signal closely mimics the clean signal’s pattern, TSD. As illustrated in Figure 6, it is evident from the Figure of our proposed algorithm for performing trojan attacks on the field.

**B. Results and Discussion**

1) **White-box Attack**: Our study showcases a consistently high fooling rate following the introduction of a trojan attack, with all 17 classes experiencing rates above 99% in Figure 4. It’s noteworthy that the decrease in accuracy was minimal, with the most significant observed decrease in baseline accuracy in class C-14, which demonstrated a reduction to 97.59%. We underscore that the high fooling rates as depicted by the confusion matrix in Figure 5, illustrate the effectiveness of our proposed algorithm for performing trojan attacks on TSD. As illustrated in Figure 6, it is evident from the Figure that the trojan signal closely mimics the clean signal’s pattern, rendering it virtually indistinguishable from a human observer in the PCC. The efficacy of our proposed algorithm is further substantiated through its consistent performance across diverse model architectures in Figure 7, as evidenced by the maximum fooling rates achieved. The algorithm exhibits remarkable fooling rates reaching 99.77% for LSTM, 99.87% for ResNet50, 99.94% for CNN-LSTM, 99.98% for ResNet18, and 99.99% for CNN. Such high fooling rates signify the algorithm’s universal applicability and effectiveness irrespective of the underlying model. Crucially, the algorithm’s sophistication is further elucidated when considering the baseline accuracy juxtaposed with the clean accuracy. Our assessment indicates that the maximum decrement in accuracy was witnessed in the LSTM model, wherein the baseline accuracy of 99.30% experienced a diminution to a clean accuracy of 98.67%, marking a drop of 0.63%. This represents the most pronounced impact among the tested architectures, yet it remains a relatively minor perturbation, showcasing the algorithm’s ability to preserve the integrity of the model’s functional accuracy to a substantial extent. We present a comprehensive evaluation of three state-of-the-art algorithms from extant literature, alongside our proposed algorithm in Figure 8. The reported average fooling rate is meticulously computed by aggregating outcomes from comprehensive experiments, encompassing all datasets explored and the application of algorithms across diverse DL models. The findings indicate that TimeTrojanDE [13] achieved a notable average fooling rate of 92.5%. TSBA [12] displayed a higher efficacy with a fooling rate of 99.07%, and TrojanFlow [11] was similarly effective, achieving a fooling rate of 99.65%. Our algorithm, ‘Sneaky Spectral Strike (S3)’, demonstrated the highest proficiency with a fooling rate of 99.90%, indicating its superior ability to subvert the models’ predictive accuracy. While the high fooling rates achieved by TSBA, TrojanFlow, and TimeTrojanDE are indicative of their effectiveness, these results were obtained on datasets with either large numbers of time steps but small sample sizes, or large sample sizes but fewer time steps. Our algorithm was tested against datasets characterized by a larger scale in both the number of samples and the extent of time steps, presenting a more challenging and arguably more realistic scenario for evaluation. It is imperative to underline this distinction because the complexity and size of a dataset can have a significant impact on the generalizability of an algorithm’s performance.

2) **Black-box Attack**: We start with rigorously evaluating the efficacy of the 'Lite Datanet Sneaky Spectral Strike' algorithm in the Resnet50 model. From Figure 9, we can observe an exceptional performance across all classes in the dataset. The high fooling rates as depicted by the confusion matrix in Figure 10, illustrate the effectiveness of our proposed algorithm for performing trojan attacks on TSD. The fooling rates ranged from 99.60% to 99.92%, with the majority of classes experiencing fooling rates above 99.50%. The algorithm’s ability to successfully fool the DL model without being detected is demonstrated in Figure 11 where
the trojan signal closely resembles the pattern of the clean signal, making it nearly indiscernible to a human observer in PCC. The clean accuracy remained robust across all classes, fluctuating slightly but generally staying above 98.80%. This aspect of our findings is crucial, as it demonstrates that the integration of the trojan trigger by our algorithm does not substantially compromise the model’s functionality on clean data. Our algorithm’s efficacy across different DL models was also evaluated in Figure 12. Across the board, the fooling rates are extremely high, with the CNN model topping the chart with a perfect 100% susceptibility, followed closely by ResNet18 at 99.99%, LSTM at 99.98%, ResNet50 at 99.87%, and CNN-LSTM at 99.44%. Despite this, the models maintain high clean accuracy rates, with ResNet18 leading at 99.88%, which is even higher than its baseline accuracy of 99.80%. Similarly, CNN-LSTM shows a slight improvement over its baseline, with a clean accuracy of 99.77% compared to 99.30%. The CNN model, while completely vulnerable to trojan insertion, still performs well on clean data with an accuracy of 99.67%.

VI. CONCLUSION

In this paper, we proposed two novel algorithms, one for white-box and another for black-box, for highly effective trojan attacks on DL models working with TSD in SG. In white-box attack, our proposed algorithm titled ‘Sneaky Spectral Strike (53)’ highlights the vulnerability of DL models to trojan attacks, as evidenced by utilizing a ResNet50 model with an accuracy rate of 99.22% on clean data, which, after the trojan attack, yielded a fooling rate of 99.87%. Our algorithm’s efficacy is evaluated by its high fooling rates across various models - a maximum of 99.99% for CNN - demonstrating its effectiveness and adaptability. In a comprehensive evaluation against three state-of-the-art algorithms, our algorithm emerged as the most proficient, achieving the highest fooling rate of 99.90% across a more complex dataset with large sample sizes and time steps. In black-box attacks, our proposed algorithm titled ‘Lite Dataset Sneaky Spectral Strike’ demonstrated exceptional efficacy in the CNN model, achieving a maximum fooling rate of 100.0%. Our algorithm effectively embedded trojans across various DL models, achieving high fooling rates, notably 99.99% for ResNet18, while maintaining impressive clean accuracies. These high fooling rates underscore the risk that trojan attacks present in SG, where even minor misclassification rates could lead to dire consequences. Furthermore, our study shows that the triggers generated by our attack are subtle rendering them hard to detect by human observers in the PCC. Future research should prioritize the development of defense mechanisms that are effective in identifying and neutralizing such threats in critical applications like SG. Additionally, investigating the adaptability and effectiveness of our proposed algorithms in diverse real-world scenarios beyond SG, particularly in other domains where TSD is crucial would be invaluable. Such research endeavors are vital for advancing cybersecurity measures in various critical infrastructures and protecting against trojan attacks.

ACKNOWLEDGMENT

The work in this project is supported by the National Science Foundation Grant number 2301553 and Cisco Grant CG#70615867.

REFERENCES