ADVERSARIAL ATTACKS AND DEFENSE MECHANISMS TO IMPROVE ROBUSTNESS OF DEEP TEMPORAL POINT PROCESSES

by

Samira Khorshidi

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy

Department of Computer and Information Science

Indianapolis, Indiana

August 2022

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. George O. Mohler, Co-Chair

Department of Computer and Information Science

Dr. Mohammad Al Hasan, Co-Chair

Department of Computer and Information Science

Dr. Rajeev R. Raje

Department of Computer and Information Science

Dr. Arjan Durresi

Department of Computer and Information Science

Approved by:

Dr. Shiaofen Fang

ACKNOWLEDGMENTS

Firstly, I would like to appreciate and thank my advisor George O. Mohler, who guided me throughout my Ph.D. program. Your impact has been wise, encouraging, and monumental. Without your invaluable assistance, I would not be able to finish this journey toward my Ph.D. degree. I want to extend my sincere thanks to my committee members, Dr. Mohammad Al Hasan, Dr. Rajeev R. Raje, Dr. Arjan Durresi, and Dr. Jeremy Carter, for their support and recommendations.

To my family, I apologize for not being there for any of the good and bad moments during the last five years. I'm incredibly grateful to you and your love, encouragement, and unfailing support of my academic venture. In particular, my mother Soraya, my father Sedigh, and my brothers Sina and Sahand. Thank you for always supporting me, even when it is not easy to do. I will forever owe my achievements to you.

Finally, I'd like to recognize all of my friends, faculty, and staff at IUPUI.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Temporal point processes (TPP) are mathematical approaches for modeling asynchronous event sequences by considering the temporal dependency of each event on past events and its instantaneous rate. Temporal point processes can model various problems, from earthquake aftershocks, trade orders, gang violence, and reported crime patterns, to network analysis, infectious disease transmissions, and virus spread forecasting. In each of these cases, the entity's behavior with the corresponding information is noted over time as an asynchronous event sequence, and the analysis is done using temporal point processes, which provides a means to define the generative mechanism of the sequence of events and ultimately predict events and investigate causality.

Among point processes, Hawkes process as a stochastic point process is able to model a wide range of contagious and self-exciting patterns. One of Hawkes process's well-known applications is predicting the evolution of viral processes on networks, which is an important problem in biology, the social sciences, and the study of the Internet. In existing works, mean-field analysis based upon degree distribution is used to predict viral spreading across networks of different types. However, it has been shown that degree distribution alone fails to predict the behavior of viruses on some real-world networks. Recent attempts have been made to use assortativity to address this shortcoming. This thesis illustrates how the evolution of such a viral process is sensitive to the underlying network's structure.

In Chapter 3, we show that adding assortativity does not fully explain the variance in the spread of viruses for a number of real-world networks. We propose using the graphlet frequency distribution combined with assortativity to explain variations in the evolution of viral processes across networks with identical degree distribution. Using a data-driven approach, by coupling predictive modeling with viral process simulation on real-world networks, we show that simple regression models based on graphlet frequency distribution can explain over 95% of the variance in virality on networks with the same degree distribution but different network topologies. Our results highlight the importance of graphlets and identify a small collection of graphlets that may have the most significant influence over the viral processes on a network.

Due to the flexibility and expressiveness of deep learning techniques, several neural network-based approaches have recently shown promise for modeling point process intensities. However, there is a lack of research on the possible adversarial attacks and the robustness of such models regarding adversarial attacks and natural shocks to systems. Furthermore, while neural point processes may outperform simpler parametric models on in-sample tests, how these models perform when encountering adversarial examples or sharp non-stationary trends remains unknown.

In Chapter 4, we propose several white-box and black-box adversarial attacks against deep temporal point processes. Additionally, we investigate the transferability of white-box adversarial attacks against point processes modeled by deep neural networks, which are considered a more elevated risk. Extensive experiments confirm that neural point processes are vulnerable to adversarial attacks. Such a vulnerability is illustrated both in terms of predictive metrics and the effect of attacks on the underlying point process's parameters. Expressly, adversarial attacks successfully transform the temporal Hawkes process regime from sub-critical to into a super-critical and manipulate the modeled parameters that is considered a risk against parametric modeling approaches. Additionally, we evaluate the vulnerability and performance of these models in the presence of non-stationary abrupt changes, using the crimes and Covid-19 pandemic dataset as an example.

Considering the security vulnerability of deep-learning models, including deep temporal point processes, to adversarial attacks, it is essential to ensure the robustness of the deployed algorithms that is despite the success of deep learning techniques in modeling temporal point processes.

In Chapter 5, we study the robustness of deep temporal point processes against several proposed adversarial attacks from the adversarial defense viewpoint. Specifically, we investigate the effectiveness of adversarial training using universal adversarial samples in improving the robustness of the deep point processes. Additionally, we propose a general point process domain-adopted (GPDA) regularization, which is strictly applicable to temporal point processes, to reduce the effect of adversarial attacks and acquire an empirically robust model. In this approach, unlike other computationally expensive approaches, there is no need for additional back-propagation in the training step, and no further network is

required. Ultimately, we propose an adversarial detection framework that has been trained in the Generative Adversarial Network (GAN) manner and solely on clean training data.

Finally, in Chapter 6, we discuss implications of the research and future research directions.

1. INTRODUCTION

Temporal point process models are employed for a variety of applications ranging from analyzing electronic transaction records [1], forecasting earthquake aftershocks [2], mitigating the spread of fake news [3], [4], and allocating police to crime hot spots [5], [6].

The current research investigates the sensitivity of point process models used for virality evolution to the underlying network structure and the robustness of neural point processes to non-stationary changes in data and adversarial attacks. Moreover, we study and propose methods to improve their robustness. Ultimately, it contributes to trustworthy and explainable artificial intelligence (AI) since, up to date, only a few studies examine the adversarial attacks against regression models, and expressly none examined point process models that can be viewed as the time-to-event regressions. Specifically, most advances in trustworthy AI focus on computer vision and image analysis. Nonetheless, point processes such as the Hawkes process are utilized in sensitive fields and security-related tasks such as criminology, fraud detection, anomaly detection, and flight arrival time prediction.

1.1 Motivation

1.1.1 Virality hinges on the underlying network's structure

A variety of dynamic phenomena, including YouTube video views [7], Tweet resharing [8], viral marketing campaigns [9], the spread of computer viruses on the Internet [10], and gang retaliation [11] can be explained as evolving viral processes on networks. As such, the study of the evolution of viral processes on networks has attracted considerable attention in recent years. It is now well known that for a connected network, the largest eigenvalue of its adjacency matrix is a good metric for predicting the viral process in that network [12]–[14]. The largest eigenvalue can be roughly estimated by the average degree of the network [15]. Still, the complete degree distribution of the network is more expressive than the point estimate of the average degree and has therefore also been considered for predicting these viral processes. A common approach along these lines is to employ a mean-field analysis where independence assumptions on the nodes are used [16]–[20]. More recently it has been

shown that in some cases, including real-world networks, these mean-field assumptions fail to predict the viral spreading [21]. Consequently, assortativity has been proposed for addressing the limitations of the degree-based, mean-field analyses [22], [23]. Through degree-preserving network rewiring procedures, it has been shown that the spectral radius of a graph can exhibit great fluctuations across networks with the same degree distribution and that assortativity can be used to explain this variation. While there is a clear correlation between assortativity and the dynamics of viral processes on graphs, the fact remains that, if we control for both assortativity and degree distribution, viral diffusion on networks with different topologies can still exhibit greatly different behaviors.

We illustrate this observation with the following example. In Figure 1.1, we display results for a Hawkes process [7] simulation on 2500 networks with identical degree distribution sampled via degree preserving rewiring [24] from the Karate network [25]. In the Hawkes branching process model, an initial event occurs at a randomly chosen node. Then subsequent generations of events occur at neighboring nodes of previous events with a fixed probability. In Figure 1.1, we plot assortativity vs. the expected total number of events (at time infinity) of the Hawkes process for each of the simulated networks. While assortativity partially explains the behavior of the process, for the two highlighted networks with identical degree distribution and similar assortativity, the expected total number of events in the process differs by a factor of 4. So, we need to extend our analysis beyond degree distribution and assortativity to understand better the dynamics of a viral process over a network.

In Chapter 3, we propose using the frequency distribution of graphlets (see Figure 1.2 for a preview) to explain the variation in viral processes observed in Figure 1.1. Specifically, we show that graphlet frequencies are good predictors for explaining the variation of the evolution of viral processes over a collection of networks for which the degree distribution is kept fixed. Our results not only highlight the importance of graphlets but also identify a small collection of graphlets that may have the highest influence over the viral processes on a network.

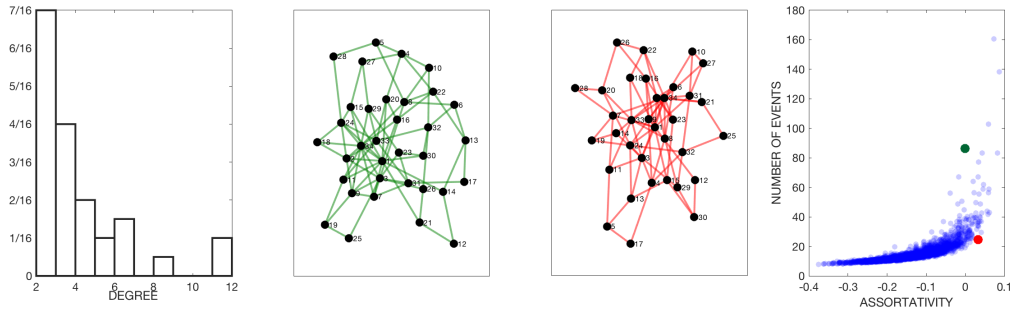


Figure 1.1. Hawkes process simulation on 2500 rewired Karate networks. Degree distribution (far left) is fixed for all of the 2500 networks. The Expected number of events in a cascade vs assortativity (far right). Two example networks (middle) corresponding to large differences in virality despite similar assortativity and identical degree distribution.

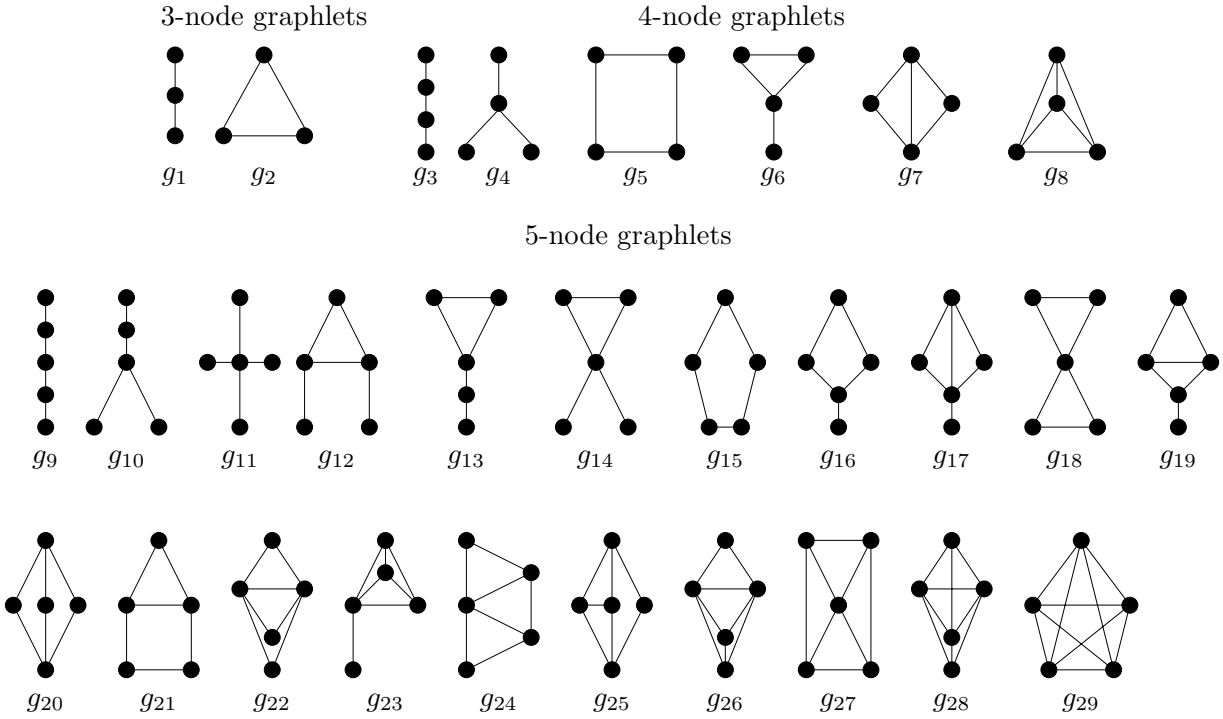


Figure 1.2. Undirected graphlets with 3, 4, and 5 vertices.

1.1.2 Adversarial attacks

Due to the success of deep learning models in various domains, neural network-based approaches to modeling point processes have recently received attention from the research community [26]–[28]. These methods attempt to learn non-parametrically key components of point processes and their intensity to capture real event patterns better than parametric models.

Despite the remarkable success of deep neural networks (DNNs), they suffer from severe vulnerabilities to adversarial attacks. As a result, vulnerability's examination of DNN in computer vision and natural language processing has received attention recently. In the point process domain, deep learning approaches run the risk of over-parameterizing models and overfitting real-world, noisy data despite their success. Furthermore, recent research on adversarial attacks has gained attention. Such research is crucial for security purposes and understanding deep learning models in order to make machine learning models trustworthy.

For neural point processes, there is a lack of research on how robust such models are to natural shocks to systems, e.g., how a pandemic impacts deep point process forecasts of crime and adversarial attacks. This study investigate neural network-based point process models and their robustness against natural shocks and adversarial attacks.

It's noteworthy that, unlike classification tasks, in regression, the adversarial learning setting is restricted by difficulties in defining the adversarial attacks, their effectiveness, and evaluation metrics [29]. In examining adversarial attacks against regression models, application-specified approaches are common. In [30] by studying attacks against autonomous driving models, the authors present the adversarial threshold, which corresponds to a deviation between the actual prediction and the prediction on the adversarial sample.

The present work is the first to explore the adversarial methods for temporal point processes modeled by deep neural networks (DNN) and examine their performance. In particular, in Chapter 4, I) We propose several adversarial attack approaches to generate white-box and black-box adversarial perturbation against point processes modeled by deep learning. In white-box settings, the network's knowledge is accessible to the adversary. In contrast, no knowledge about the attacked network is required in black-box attacks. And

II) We study the transferability of white-box attacks where the adversarial samples are crafted on networks with more significant parameters and employed to attack the targeted network. Transferability of adversarial attacks increases the risk of adversarial attacks. III) We show how adversarial attacks can disturb underlying parameters of point processes that are considered a threat to parametric modeling.

In Figure 1.3 and Figure 1.4, we present the overview of white-box attacks and black-box adversarial attacks respectively. Here, original points are illustrated in Blue, while Red represents adversarial points that are the input to the model. Moreover, we present the model's prediction for the original and corresponding adversarial input sequence in Green and Orange, respectively. The affiliated intensity of each point has been shown to emphasize the effect of adversarial attacks on underlying parameters.

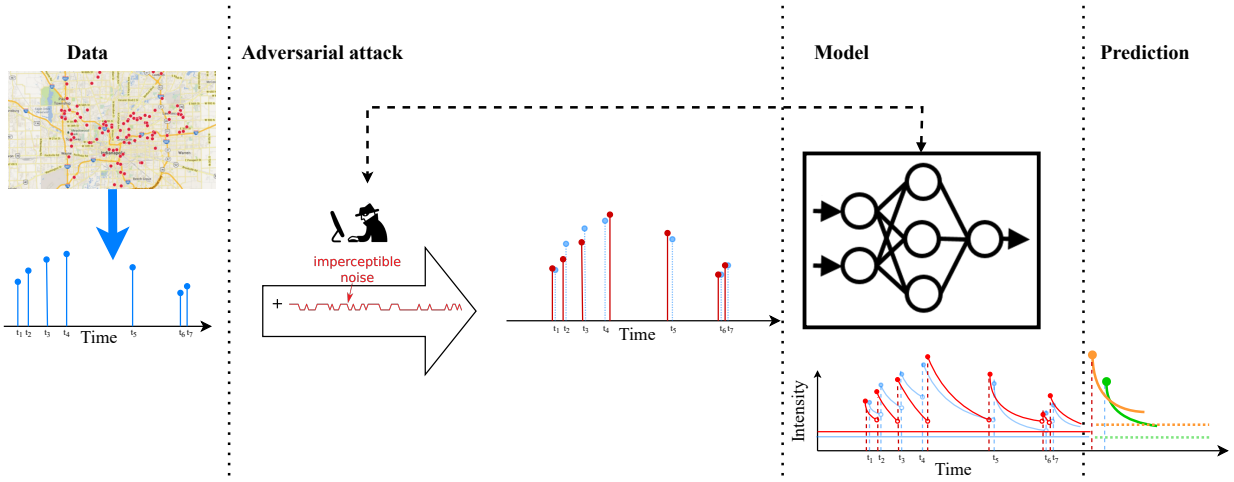


Figure 1.3. The overview of proposed *white-box* adversarial attacks against neural point processes. In this framework, temporal components of points, e.g., crimes, are the input of the DNN model. The adversary has some knowledge about the network and employs it to generate adversarial samples.

1.1.3 Robustness to adversarial attacks

As aforementioned, adversarial attacks against deep learning received boosted attention in recent years. Consequently, for all machine learning models, including neural networks,

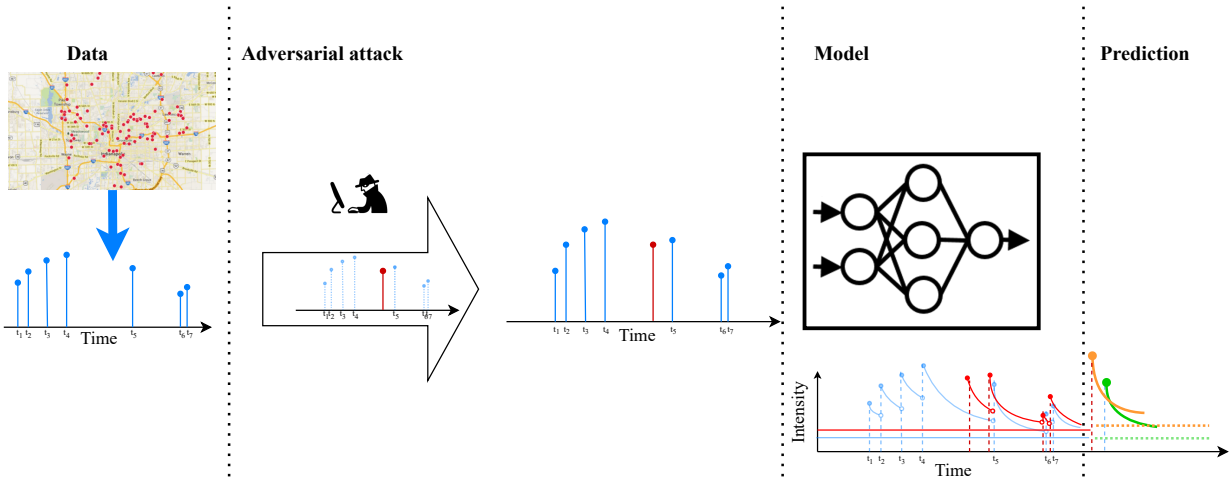


Figure 1.4. The overview of proposed *black-box* adversarial attacks against neural point processes. In this framework, temporal components of points, e.g., crimes, are the input of the DNN model. Unlike white-box attacks, in black-box attacks, the adversary introduces a fake point (time) independent of the network architecture and weights.

adversarial robustness and defense against adversarial attacks are essential for security purposes.

In Chapter 5, we investigate the robustness of neural point processes against several proposed adversarial attacks. Specifically, from the adversarial defense viewpoint, adversarial attack strategies aim to perturb original real data samples that can be systematically adopted to design defense mechanisms, e.g., adversarial learning. In Chapter 5, I) We investigate the effectiveness of adversarial training using universal adversarial samples. II) We propose domain adopted loss function for neural point processes based on point process residual analysis theorem to improve the robustness against several adversarial attacks. Finally, III) We propose an adversarial detector that has been trained in Generative Adversarial Network (GAN) frameworks and solely on clean training data.

In this research, we extend the topic of trustworthy machine learning to point processes and sequential data. Specifically, we step towards making neural point process models explainable, detect their weaknesses, and robustness. Additionally, due to advances in deep learning, we examine and propose several adversarial attacks against point processes in general and specific fields. Finally, regarding robustness against adversarial attacks, we experiment theoretical approaches to improve the model robustness empirically by proposing a simple, novel, and flexible regularization methods.

2. BACKGROUND

In this research and for Chapter 3, we will be making several different measurements reflecting the topology of networks from degree distribution to assortativity to graphlet distribution – as well as simulating two different viral processes – the Hawkes process and Susceptible-Infected-Susceptible model – on networks. We provide some background materials on these topics in this section.

2.0.1 Degree Distribution

The *degree* of a node is the number of connections of that node to other nodes in the network. The degree distribution is a discrete probability distribution of degrees over the nodes in the network. Directed networks have two different degree distributions, the in-degree and the out-degree distributions. In this study, we restrict our attention to undirected networks for which only one degree distribution is defined.

2.0.2 Assortativity

Assortativity, or assortative mixing, is defined by the tendency of a network's nodes to be connected to others nodes that are similar in some way. While there are several different mathematical definitions, we will refer to assortativity as the Pearson correlation coefficient of degrees at either ends of a network edge [31]. In this case the assortativity is given by the formula [31],

$$A = \frac{M^{-1}\sum_{\mathrm{i}} \mathrm{j}_{\mathrm{i}} k_{\mathrm{i}} - [M^{-1}\sum_{\mathrm{i}} \frac{1}{2}(\mathrm{j}_{\mathrm{i}} + k_{\mathrm{i}})]^2}{M^{-1}\sum_{\mathrm{i}} \frac{1}{2}(\mathrm{j}_{\mathrm{i}}^2 + k_{\mathrm{i}}^2) - [M^{-1}\sum_{\mathrm{i}} \frac{1}{2}(\mathrm{j}_{\mathrm{i}} + k_{\mathrm{i}})]^2}, \tag{2.1}$$

where the edges in a network are indexed by $\mathrm{i} = 1, ..., M$ and $\mathrm{j}_{\mathrm{i}}, k_{\mathrm{i}}$ are the degrees of the nodes at the ends of edge i. In Figure 2.1, we provide examples of two social networks from the Network Repository [25] with different assortativity, one positive and one negative.

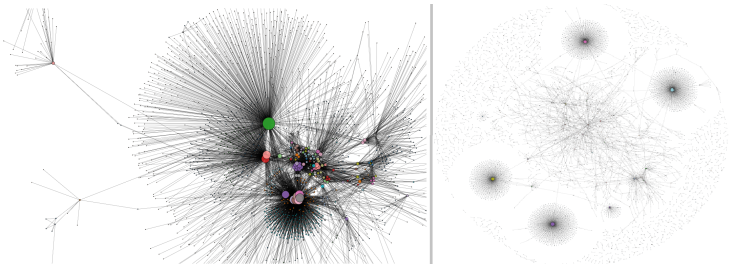


Figure 2.1. The left figure shows a graph with positive assortativity where nodes with high degree are connected to other nodes with high degree ($assortativity = 5.4$). The right graph is an example of a disassortative network,($assortativity = -0.88$) where the hubs connect to low degree nodes.

2.0.3 Graphlets

Graphlets can be defined as small, connected [1], non-isomorphic, induced subgraphs of a large network. In this study, we work with all possible graphlets having $k \in \{3, 4, 5\}$ vertices. If the graphlet edges are undirected, there are 29 such graphlets as shown in Figure 1.2. We refer to a graphlet with k vertices as a k-*Graphlet*; note that a 1-*Graphlet* is simply a vertex and a 2-*Graphlet* is simply an edge.

The frequency of a graphlet g_{i} in a graph G is the total number of distinct embeddings of that graphlet g_{i} in the graph G. The Graphlet frequency distribution (GFD) is the normalized frequency of the graphlets. To obtain GFD, we normalize the graphlet count vector (a vector of 29 integers which represents the count of each of the graphlets) so that the $L1$-norm of the vector is 1; thus, the values in the vector represent a discrete probability distribution.

As real life networks are generally sparse, frequencies of larger sized graphlets shrink exponentially, so in GFD, we typically use a logarithm scale for comparing various frequencies so that the number of occurrences of larger-sized graphlets against that of the smaller-sized ones are scaled appropriately.

The frequency distribution of graphlets in a network captures the local topology around the vertices of the network and it has a number of uses in network analysis and prediction.

[1]↑disconnected graphlets have also been considered in several graphlet based works, but in this work we only consider connected graphlets because connectedness is essential for the evolution of a viral process on a network.

For example, the frequencies of various graphlets can be used for building a global fingerprint for a network such that the fingerprints of different networks arising from a real-life domain are almost identical [32]. Transition of graphlets over temporal snapshots of a network has also been shown to improve link prediction models for dynamic networks [33].

A key challenge for finding graphlet frequency distribution is the high computational cost of graphlet enumeration or counting. However, in recent years, several efficient algorithms have become available that generate exact graphlet counting [32], [34]. Yet, for very large networks exact counting of graphlets is not feasible. So, there exist methods (such as, GUISE and its variants [35]–[37]), which can generate approximate graphlet frequency distributions through uniform sampling of graphlets. Sampling based methods provide very good approximations of graphlet frequency distributions and can scale to graphs with millions of vertices. In this work, we use GUISE algorithm for computing the graphlet frequency distributions of a network.

2.0.4 Hawkes process model

The Hawkes process is a specific kind of self-exciting point process whereby discrete events occur according to a stochastic intensity $\lambda(t)$ that increases in response to events themselves. For a Hawkes process on a network G, given the event observation dyads $(t_{\mathrm{i}}, v_{\mathrm{i}})$, where t_{i} is a time value for an event and v_{i} is a vertex on which the event occurred, the conditional intensity (rate) of events at node v, $\lambda_v(t)$, is given by the equation

$$\lambda_v(t) = \mu + \sum_{\substack{t>t_{\mathrm{i}} \\ v_{\mathrm{i}} \in N(v)}} \theta f(t - t_{\mathrm{i}}). \tag{2.2}$$

In (2.2), $N(v)$ is the set of nodes that are neighbor of v on G (note that $v \in N(v)$ for this process) so that the intensity is a superposition of a Poisson background intensity μ and Poisson intensities $\theta f(t - t_{\mathrm{i}})$ centered at previous event times t_{i} such that v_{i} is a neighbor of v on the graph G. The triggering kernel $f(t)$ is a probability density defined on $[0, \infty)$ and, when the model is interpreted as a branching process, the productivity parameter θ

determines the expected number of direct offspring events at node v triggered by an event at a node $v_{\mathrm{i}} \in N(v)$.

2.0.5 Susceptible-Infected-Susceptible model

The second model we consider is based on the well-known Susceptible-Infected-Susceptible (SIS) model in epidemiology. In this model, each node of the network can be in one of two states at any given time - susceptible or infected. Let $S(t)$ be the set of all susceptible nodes at time t, $I(t)$ be the set of all infected nodes at time t, $\lambda_v(t)$ be the rate at which a susceptible node $v \in S(t)$ becomes infected, and μ_u be the rate at which an infected node $u \in I(t)$ becomes susceptible. The model is then

$$\lambda_v(t) = \sum_{q \in N(v) \cup I(t)} \theta \tag{2.3}$$

$$\mu_u = 1 \ . \tag{2.4}$$

So, a susceptible node v switches to being infected via a Poisson process with time varying rate equal to a parameter θ times the number of neighbors of v that are infected at that time, and an infected node u switches to being susceptible via a homogeneous Poisson process with rate 1 (without loss of generality). In terms of virality, one may be interested in a scenario in which all nodes are initially susceptible except for a potentially small number of infected, then tracking how many further infections occur as a result of these initial infections. The result will clearly depend on which nodes are initially infected, the adjacency matrix of the network A, and the parameter θ.

3. THE ROLE OF GRAPHLETS IN VIRAL PROCESSES ON NETWORKS

3.1 Methodologies

Our primary objective is to show that graphlet frequency distribution, in addition to assortativity, is a good predictor for the virality in a network when degree distribution is controlled for. Many earlier works use analytic approaches for finding the influence of network topology or network based metrics on the viral process on a network [12], [13]. However, such methods are very cumbersome for graphlets, as graphlets are combinatorially complex objects and their influence over the dynamic process is difficult to represent by a simple model that can be solved analytically. So, in this work we forgo a mathematical analysis in favor of a data science approach to the problem. [1]

Our overall strategy is to use simulation and empirical measurement to explore the connection between graphlet distribution and the evolution of viral processes on networks. For this purpose we use several real-world networks as input to a simulation model. Given a particular network and model for a viral process, we perform the following steps:

1. Generate M synthetic networks through re-wiring (explained below) with identical degree distributions to the original real-world network.

2. For each generated network, compute the assortativity and graphlet frequency distribution.

3. For each generated network, compute the expected number of events, $E[N_\infty]$, of the viral process of interest running on the network.

4. Regress $E[N_\infty]$ against the assortativity and/or graphlet frequency distribution to assess the role that these measures play beyond degree distribution.

Below we provide more details of the above steps.

[1]↑Reproduced with permission from Springer Nature

3.1.1 Generating Simulation Graphs

We have shown in the Introduction section, degree distribution and assortativity are not adequate for explaining the evolution of a viral process in a network—which motivate us to find the influence of graphlet frequency distribution in a viral process. However, degree distribution does provide a partial explanation of a viral process. To nullify the influence of degree distribution in our analysis, we use degree distribution as a control variable, i.e., we generate a collection of synthetic networks for which degree distribution is a constant.

Generating networks with a given degree distribution is a well-studied problem, specifically for the task of network motif discovery [38]. There are two well-known approaches for solving this problem, (i) edge swapping [39], [40] and (ii) stub-matching. Edge swapping starts from a given graph and makes local modification on the given graph to generate another graph having the same degree sequence. One edge swapping approach that preserves the degree sequence is the following. First, select two edges uniformly at random from the graph G; for example, suppose these are $e_1 = (a, b)$ and $e_2 = (c, d)$. Then, replace these two edges by two new edges where the second vertices are swapped between the original two edges, assuming those new edges are not already present in G; in our example, these would be the two new edges $e_3 = (a, d)$ and $e_4 = (c, b)$. If the edge e_3 or e_4 (or both) already exists in G, this proposed swap is rejected and the process is repeated with a new pair of randomly chosen edges e_1 and e_2. It is easy to see that the degree of each vertex remains invariant under a successful edge swap process. The edge-swapping can be continued and a sequence of graphs can be generated, such that all of these graphs have an identical degree distribution.

If we consider the sequence of graphs as a Markov chain, then the stationary distribution of the Markov chain is a uniform distribution over the graphs having identical degree distribution. For stub-matching, the configuration model is very popular [41]. In this method, the algorithm creates as many stubs (dangling half-edges) for each vertex as its degree. Then edges are created by choosing pairs of vertices randomly and connecting their stubs. This approach may create parallel edges, which are dealt with by restarting the process; for large graph the re-starting may become very costly.

In this work, we use the edge-swapping method, as it is easy to implement. By choosing a sufficiently large number of steps for the Markov chain, we generate graphs which are sufficiently different from each other with widely different graphlet frequency distributions.

3.1.2 Preparing Topology and Virality Data for Regression

For our study, a collection of graphs with identical degree distribution is a regression dataset in which each graph is an instance. For each graph we compute graphlet degree distribution and assortativity, which become the explanatory variables for our regression. Below we discuss how we compute these values for a given graph.

Computing graphlet frequency distribution by counting each of the graphlets in a graph is a costly task as the number of graphlet embeddings grows exponentially with the size of the graphlets. In fact, if both connected and disconnected graphlets are considered, the number of k-graphlets on a graph with K vertices is equal to $O(\binom{K}{k})$. In this work, we consider graphlets up to size 5, for which a brute-force graphlet enumeration complexity is equal to $(O(K^5)$, which is not scalable for many real-life networks. An alternative to counting is uniform sampling of graphlet embeddings, which is sufficient to obtain a graphlet frequency distribution. In an earlier work [34], we have shown how graphlets can be sampled under uniform distribution by using a Monte Carlo Markov Chain (MCMC) sampling algorithm.

Specifically, we have proposed a method named GUISE, which performs a random walk over the graphlet embeddings by following a double-stochastic transition matrix; the stationary distribution of the Markov chain is a uniform distribution over the graphlet embeddings. By counting the type of graphlets that are traversed in this random walk and then normalizing the vector as described above, GUISE returns a graphlet frequency distribution vector. In this work, we use GUISE algorithm for computing the graphlet frequency distribution. It returns a 29-size vector, in which each component $d(g_i)$ represents the normalized frequency of the graphlet g_i as illustrated in Figure 1.2. We also compute the assortativity of the network using (2.1). The components of the graphlet frequency distribution vector and assortativity (a 30-size vector) become the covariates of our regression analysis.

The target value of our regression is the expected number of events – excited offspring events in the case of the Hawkes process and secondary infections in the case of the SIS process – that are spawned from a single initiating event placed randomly within the network.

The way this target value is computed depends on the viral process used. For the Hawkes process [7], [8], [11], we consider a simplified model where

1. a node is chosen uniformly at random
2. an initial event at time $t_1 = 0$ occurs at the chosen node
3. the Hawkes process with $\mu = 0$ is simulated and the total number of events, N_∞, at time infinity is observed

In the case of this simplified model, the expected number of total events is given by,

$$E[N_\infty] = \frac{1}{K} 1^T \cdot (\sum_{\mathrm{j}=1}^{\infty} (\theta A)^{\mathrm{j}} \cdot 1), \tag{3.1}$$

where K is the number of nodes in G, 1 is a column vector of ones, and A is the adjacency matrix of G (A is symmetric because the graph G is undirected). The expected total number of events $E[N_\infty]$ will be finite up to a critical threshold value of the productivity parameter, θ_c.

For the SIS model, we approximate the continuous time version described in Section 2 above with a discrete time version that allows us to more easily count the number of secondary infections arising from a single initially infected node, and which greatly simplifies the simulations. Here, time is discretized into units of step 1, and we define a vector $I(t)$ such that $I_u(t) = 1$ if node u is infected at time t and 0 if node u is susceptible at time t. Initially, $I_u(0)$ is zero for all u except for a single node chosen uniformly randomly. Then the model proceeds via iterations of the following steps:

1. nodes v susceptible at time t become infected at time $t+1$ with probability $1 - \mathrm{e}^{-\lambda_v(t)}$, where $\lambda_v(t) = \theta(AI(t))_v$
2. all nodes infected at time t become susceptible at time $t+1$

3. at each timestep $t > 0$, the product $1^T \cdot I(t)$ is equal to the new number of infections

4. the total new infections N_∞ are observed as time goes to infinity

For this simplified model, an approximation of the expected number of total new infections can be found via

$$E[N_\infty] = \frac{1}{K} 1^T \cdot \sum_{k=1}^{K} \sum_{t=1}^{\infty} I(t;k) \tag{3.2}$$

$$I_v(0;k) = \delta_{v,k} \tag{3.3}$$

$$I_v(t+1;k) = (1 - I_v(t;k)) \left[1 - \mathrm{e}^{-\theta(AI(t;k))_v}\right] \ . \tag{3.4}$$

As in the Hawkes model, the value of $E[N_\infty]$ is expected to be finite up until some critical value of θ, below which the infection is expected to disappear at some finite time (is at most epidemic), and above which the infection in expectation never leaves the network (is endemic).

3.1.3 Regression Model for Predicting Virality

For each real-world network we simulate $M = 500$ rewired networks holding the degree distribution fixed. Next, for each simulated network, we compute the assortativity, the graphlet frequency distribution, the largest eigenvalue λ_{max}, as well as $E[N_\infty]$ for the Hawkes and SIS models. When calculating $E[N_\infty]$ for the Hawkes and SIS models, the value of θ is chosen to be a constant multiple of the largest eigenvalue of the original network. To explain the observed variation in $E[N_\infty]$ across the simulated networks, we run a regressions of the form

$$\log(E[N_\infty]) = b + c_0 A + \sum_{\mathrm{i}=1}^{29} c_{\mathrm{i}} \log(d(g_{\mathrm{i}})) + \epsilon \tag{3.5}$$

where b is the intercept, A is the assortativity, $d(g_{\mathrm{i}})$ is the frequency distribution value of graphlet g_{i}, and c_{i} are coefficients of a linear regression where the model errors ϵ are assumed to be normal. Note that, although our regression is simply a linear fit, the underlying relationship between virality and the various graph topology measures is proposed to be nonlinear due to the logarithms present in (3.5). This nonlinearity is certainly plausible at

Table 3.1. Network statistics **as reported by** [25].

	rt-retweet	karate	soc-dolphins	soc-firm-hi-tech
Nodes	96	34	62	33
Edges	117	78	159	124.5
Density	0.0257	0.1390	0.0841	0.2358
Maximum degree	17	17	12	28
Minimum degree	1	1	1	0
Average degree	2	4	5	9.19
Assortativity	-0.1792	-0.4756	-0.0436	-0.1200
Number of triangles	36	135	285	454.5
Average number of triangles	0	3	4	13.77
Maximum number of triangles	6	18	17	88.5
Average clustering coefficient	0.0608	0.5706	0.2590	0.4050
Fraction of closed triangles	0.0742	0.2557	0.3088	0.2960
Maximum k-core	4	5	5	8

least for the assortativity, given the plotted relationship in Fig. 1.1. Further, as discussed above, it is natural to consider the logarithm of the graphlet frequency distribution, which is why we do so here. We estimate the model on 70% of the 500 simulated networks and then evaluate the R^2 and mean square error (MSE) on the remaining 30% test data.

3.2 Data and Results

We consider four real-world networks obtained from the Network Repository [25]. The networks include 1) a retweet network where the nodes are twitter users and edges are retweets (collected from various social and political hashtags); 2) a karate network where the dataset contains social ties among the members of a university karate club collected by Wayne Zachary in 1977; 3) a social network of bottlenose dolphins where the dataset contains a list of all the links (a link represents frequent associations between dolphins); and 4) a social network from a high tech firm where no description is available on the network repository. **The statistics for the four networks as reported by the Network Repository are provided in Table 3.1**. The θ values used in the viral processes for the networks are: retweet $\theta = .98\lambda_{max}$, karate $\theta = .96\lambda_{max}$, dolphins $\theta = .95\lambda_{max}$, firm-hi-tech $\theta = .99\lambda_{max}$.

In Figure 3.1 we plot $E[N_\infty]$ vs. assortativity for the rewired versions of each of our four networks. There is clearly a positive but nonlinear relationship between virality and assortativity. However, for fixed assortativity (and fixed degree distribution by design) the virality models produce $E[N_\infty]$ values differing by an order of magnitude between network rewirings in some cases. This highlights the need for further explanatory variables to explain the virality, such as our proposed graphlet frequency distribution. As an important first check as to whether graphlet frequency distribution might possibly play an important role, we also plot in Figure 3.1 the variation in graphlet frequency distribution across the rewired networks, observing that there can be significant differences in graphlet frequency distribution between network rewirings. We emphasize here that the graphlet frequency distribution is a measure that is made on each individual rewiring, and that the plot in Figure 3.1 summarizes these various distributions for all of our simulated rewirings. So, for example, $g28$ in network soc-firm-hi-tech displays a small value within the graphlet frequency distribution, meaning that in any given rewiring, there are relatively few of these graphlets present. However, noting that the spread of values for this graphlet on the plot does not include 0, it can be concluded that every rewiring of this network displayed this graphlet to some extent. Since the box plot displays non-trivial variations of graphlet frequency distribution between network rewirings, it is at least possible that the graphlet frequency distribution could be a contributing factor to the variance of $E[N_\infty]$ observed at fixed assortativity in Figure 3.1.

To verify whether graphlet frequency distribution does in fact play an important role in virality, in Table 3.2 we provide the results for a nested regression predicting the Hawkes virality statistic where assortativity or graphlets alone are used, compared to the full model ((3.5)) with both assortativity and graphlets. We observe that the R^2 values when using only graphlets are slightly larger than when using only assortativity for all networks but soc-dolphins, where the R^2 for graphlets alone is considerably smaller than that of assortativity alone. We observe that the R^2 values increase by around 10% over assortativity alone when graphlets are also considered, and in all cases but one, including graphlets with assortativity allow for over 90% of the variance to be explained. The mean square error also improves with the addition of graphlets to assortativity in the model, with the improvement being a factor of 2 to 4. In Table 3.3 and 3.4 we provide the analogous results for a nested regression predicting

the SIS statistic and largest eigenvalue (respectively). Here we see similar improvements when the graphlets are added to the regression model over assortativity alone, and find that graphlets alone compare to assortativity alone in a similar manner as in the Hawkes model.

Table 3.2. Model comparison for predicting log number of events for Hawkes.

	Assort		GFD		Assort+GFD	
Network	R^2	MSE	R^2	MSE	R^2	MSE
Rt-retweet	7.22E-01	1.27E-01	7.81E-01	9.96E-02	8.62E-01	6.31E-02
Karate	8.14E-01	3.95E-02	8.97E-01	2.19E-02	9.63E-01	7.94E-03
soc-dolphins	8.65E-01	3.03E-04	7.12E-01	6.43E-04	9.04E-01	2.15E-04
soc-firm-hi-tech	8.54E-01	2.33E-03	8.70E-01	2.08E-03	9.33E-01	1.07E-03

Table 3.3. Model comparison for predicting log number of events for SIS.

	Assort		GFD		Assort+GFD	
Network	R^2	MSE	R^2	MSE	R^2	MSE
Rt-retweet	8.36E-01	1.32E-02	8.58E-01	1.14E-02	9.28E-01	5.78E-03
Karate	8.77E-01	2.91E-03	9.08E-01	2.17E-03	9.76E-01	5.59E-04
soc-dolphins	8.86E-01	1.47E-04	7.05E-01	3.80E-04	9.16E-01	1.08E-04
soc-firm-hi-tech	8.93E-01	5.24E-04	8.78E-01	6.00E-04	9.57E-01	2.09E-04

Table 3.4. Model comparison for predicting log of largest eigenvalue.

	Assort		GFD		Assort+GFD	
Network	R^2	MSE	R^2	MSE	R^2	MSE
Rt-retweet	8.51E-01	4.21E-05	8.98E-01	2.87E-05	9.50E-01	1.41E-05
Karate	8.66E-01	9.74E-06	9.14E-01	6.28E-06	9.78E-01	1.61E-06
soc-dolphins	8.74E-01	3.98E-06	7.21E-01	8.80E-06	9.15E-01	2.69E-06
soc-firm-hi-tech	9.25E-01	2.82E-06	8.77E-01	4.62E-06	9.72E-01	1.04E-06

Next we inspect the statistical significance of the regression coefficients to better understand which graphlets are predictive of $E[N_\infty]$ and λ_{max}. In Table 3.5, we list the independent variables in (3.5) that are significant at the .01 level. For the rt-retweet and soc-dolphins networks, graphlets lower than g8 are never selected and it appears that larger graphlets are needed to improve the model beyond assortativity. On the other hand, lower order graphlets

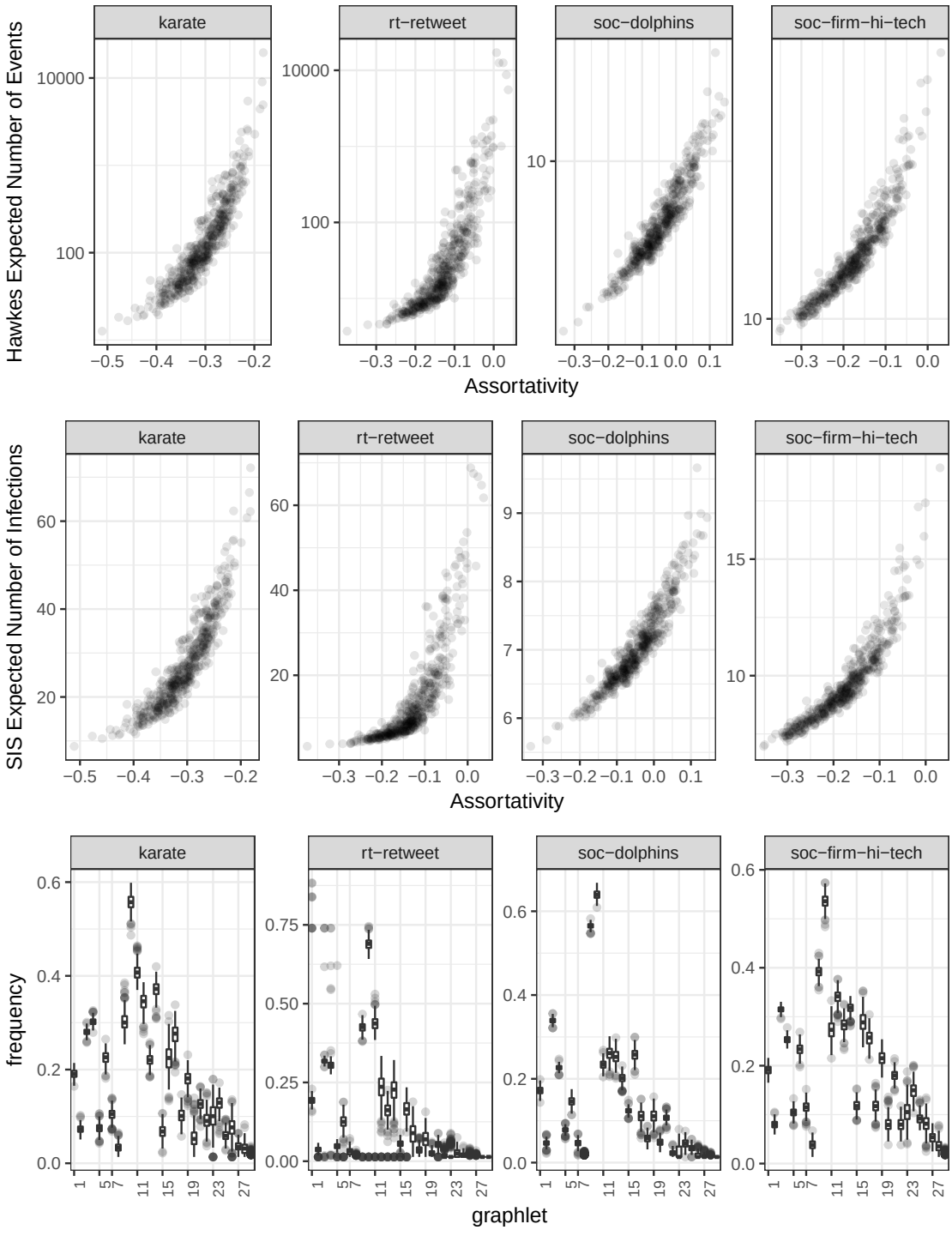


Figure 3.1. Top: Hawkes $E[N_\infty]$ vs. assortativity in the 500 rewired networks for each real-world network. Middle: SIS $E[N_\infty]$ vs. assortativity. Bottom: Box plot of graphlet frequency distributions $d(g_\mathrm{i})$ across the 500 simulated networks for each real-world network.

are significant for the soc-firm-hi-tech network, where triangles are significant across the viral process models.

To improve the regression model in (3.5), we consider an interaction model where the statistically significant variables from Table 3.5 are used and interaction terms of the form $A \cdot \log(d(g_i))$ are added. In Table 3.5 we display the R^2 values for this interaction model. In some cases we see large improvements, for example in the case of the retweet network and the Hawkes model the R^2 value increases from .86 to .95 (the R^2 for assortativity alone is .72). In the majority of cases the R^2 value of this interaction model is above .95 and for the karate model is above .98. The R^2 value for the soc-dolphins network is slightly lower, .9 to .92. Given that only high order graphlets are selected in the soc-dolphins network, it may be the case that graphlets beyond $g_2 9$ are needed to achieve R^2 values close to 1 for that network.

Table 3.5. Important variables along with R^2 values for interaction regression model.

	network	important variables (.01 level)	R^2
Hawkes	rt-retweet	A,g8,g10,g13,g18,g21,g22,g24,g25,g27	0.950
	karate	A,g1,g3,g4,g6,g9,g10,g11,g12,g13,g14,g16,g17,g18,g19,g28	0.983
	soc-dolphins	A,g10,g12,g25,g26,g28	0.913
	soc-firm-hi-tech	A,g1,g3,g4,g6,g9,g10,g11,g12,g13,g14,g16,g17,g18,g19,g23,g27,g28,g29	0.977
SIS	rt-retweet	A,g10,g11,g14,g15,g16,g17,g19,g20,g21,g23,g24,g25	0.969
	karate	A,g3,g10,g12,g14,g15,g18,g19,g20,g24,g26,g27,g28	0.986
	soc-dolphins	A,g21,g24,g26	0.902
	soc-firm-hi-tech	A,g3,g8,g10,g12,g15,g19,g25,g27,g28	0.972
λ_{max}	rt-retweet	A,g10,g13,g18,g21,g22,g24,g25,g27	0.940
	karate	A,g10,g12,g13,g14,g15,g18,g19,g20,g23,g24,g25,g26,g27,g28	0.984
	soc-dolphins	A,g10,g12,g14,g21,g26	0.929
	soc-firm-hi-tech	A,g1,g3,g4,g6,g9,g10,g11,g12,g13,g14,g16,g17,g18,g19,g23,g28	0.977

4. ADVERSARIAL ATTACKS AGAINST DEEP TEMPORAL POINT PROCESSES

4.1 Related works

Temporal point processes are practical mathematical tools for modeling event data in which the inter-event times as a random variable are modeled. Therefore, there is no required time window to aggregate events, which may cause discretization errors; this is the main difference between point process models and the discrete-time representation utilized in time series analysis [42]. Moreover, point processes can be deterministic or stochastic, and non-stationary Poisson, self-correcting, and Hawkes process are stochastic point processes that we have utilized in the current work.

As a result of advances in deep learning techniques, researchers have proposed RNN to model the intensity function of point processes [26], [43]. Most of the proposed methods utilize Long-Short Term Memory (LSTM) [44]; that we reiterate its formulation as follows

$$
\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{\mathrm{i}}\mathbf{x}_t + \mathbf{U}_{\mathrm{i}}\mathbf{h}_{t-1} + \mathbf{V}_{\mathrm{i}}\mathbf{c}_{t-1} + \mathbf{b}_{\mathrm{i}}),\\
\mathbf{f}_t &= \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{V}_f\mathbf{c}_{t-1} + \mathbf{b}_f),\\
\mathbf{c}_t &= \mathbf{f}_t\mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c),\\
\mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{V}_o\mathbf{c}_t + \mathbf{b}_o),\\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned}
\tag{4.1}
$$

where $\odot$ is element-wise multiplication, and σ is the logistic sigmoid function. The above system can be abstract as an LSTM equation as $(\mathbf{h}_t, \mathbf{c}_t) = \mathrm{LSTM}(\mathbf{x}_t, \mathbf{h}_{t-1} + \mathbf{c}_{t-1})$ [26].

Such a model can encode a nonlinear link between the predicted transient occurrence intensity of different types of events with the history of participator events, which is more expressive, and it can model more complexity in data than previous parametric or non-parametric models[45].

The RNN-based model proposed in [26] provides a black-box approach to model the intensity while both background and effect of history are considered. In their modeling, presented in (4.2) below, the background intensity is modeled by an RNN as a time series

with its units aligned with time series indexes, $\{y_t\}_{t=1}^{T}$, while another RNN handles the historical events whose units are aligned with asynchronous events to capture the long-range dynamics, $\{z_{\mathrm{i}}, t_{\mathrm{i}}\}_{t=1}^{N}$.

$$
\begin{aligned}
(\mathbf{h}_t^y, \mathbf{c}_t^y) &= \mathrm{LSTM}_y(\mathbf{y}_t, \mathbf{h}_{t-1}^y + \mathbf{c}_{t-1}^y), \\
(\mathbf{h}_t^z, \mathbf{c}_t^z) &= \mathrm{LSTM}_z(\mathbf{z}_t, \mathbf{h}_{t-1}^z + \mathbf{c}_{t-1}^z), \\
\mathbf{e}_t &= \tanh(\mathbf{W}_f[\mathbf{h}_t^y, \mathbf{h}_t^z] + \mathbf{b}_f), \\
\mathbf{U}_t &= \mathrm{softmax}(\mathbf{W}_U \mathbf{e}_t + \mathbf{b}_U), \\
\mathbf{u}_t &= \mathrm{softmax}(\mathbf{W}_u[\mathbf{e}_t, \mathbf{U}_t] + \mathbf{b}_u), \\
s_t &= \mathbf{W}_s \mathbf{e}_t + b_s,
\end{aligned}
\tag{4.2}
$$

In (4.2), the subscripts U and u indicate the primary type and sub-type of events, and s is the timestamp of each event.

Besides achievements in neural networks and their applications, recent research has revealed that neural network models in practice are vulnerable to misclassifying adversarial samples that have been crafted by adding an imperceptible additive perturbation to the data samples. In deep learning models, such a vulnerability was assumed to be explained by nonlinearity and overfitting. However, [46] argues such an assumption and shows deep learning models are vulnerable because of the linearity of adversarial perturbations, which can be analyzed as a property of high-dimensional dot products. On the other hand, neural networks, e.g., ReLU, and LSTM networks, are overly linear to oppose linear adversarial perturbation. In their work, they have suggested fast methods to generate adversarial samples, such as the fast gradient sign method (FGSM) [46], and claimed in adversarial sample creation, the perturbation's direction is more important than the specific point in space.

In addition to security concerns, research in the robustness of deep learning models are showing study of adversarial examples crafted under limited situations is helpful since it provides new insight into the geometrical characteristics and behavior of models in high-dimensional space; e.g., the characteristics of adversarial images close to the decision boundaries can help describe the boundaries' shape[47].

Recent research experiments have shown the amount of perturbation to fool deep learning models can be extremely limited, as [48] proposed a low cost, black-box attack to fool

visionary deep learning models, where the only available information is the probability labels and only one pixel can be modified based on differential evolution (DE). One critical property of such attacks is their flexibility; they can attack different networks regardless of their differentiability.

Generally, adversarial attacks are not limited to visionary, and speech-to-text systems are also exposed to misclassifying adversarial samples. Reference [49] examines the adversarial attacks in the audio domain using the Connectionist Temporal Classification Loss Function (CTCLF) as an attack mechanism and PCA as an attack and defense mechanism. In this experiment, CTCLF and PCA, as black-boxed approaches, have successfully attacked DeepSpeech[1]. In contrast, PCA as a defense mechanism does not improve the performance of DeepSpeech against adversarial attacks.

Point processes and viral processes are confirmed to be sensitive to changes in network structure. Reference [50] has shown the evolution of viral processes on a network is highly sensitive to the structural features of the network. They have discussed that assortativity and degree distribution cannot fully explain the variance in the spread of viruses; instead, graphlet distribution can explain such a variance.

Additionally, Hawkes process is used in anomaly detection. For example, reference [51] proposes a framework using the multivariate Hawkes process and reinforcement learning as a fake news mitigation framework on networks. The point process defines "mitigation" in this work on the network, and finding the optimal mitigation strategy is the objective that determines how to adjust the exogenous intensity of the few mitigator nodes on the network.

In today's electronic life, people use smartphones to monitor their activities, sleep, and health. Dealing with missing data is a new issue that arises while having augmented medical applications under partially observed Event Streams. To impute missing events, in 2019, Mei et al. [52] proposed a general sequential Monte Carlo (SMC) method using the point process, which approximates the posterior distribution over partially observed draws from a neural point process, specifically neural Hawkes process [53]. The developed method utilizes bidirectional continuous-time LSTM that allows the proposals to be conditional on future and past observed events. Additionally, they have proposed a new metric, "an optimal transport

[1]↑Speech-to-text neural network implemented by Mozilla

distance between event sequences", that measures an event sequence as a 0-1 function over times and employs a variant of Wasserstein distance [54] or Earth Mover's distance [55]–[57].

In time series area, there are two type of anomalies, point anomaly and collective anomaly. Anomalies can be detected through thresholding which is a simple filter, however they are unable to detect contextual anomalies. Another approach is Statistical Process Control which data points are considered as anomaly if they failed to pass the statistical hypothesis test. Another set of approaches are unsupervised machine learning approaches such as clustering and finding outliers.

It's not surprising that deep learning methods are also proposed to both generate and detect anomalies in time series. In 2020, TadGAN [57], Time Series Anomaly Detection Using Generative Adversarial Networks, has been proposed, which utilizes cycle-consistent GAN to detect anomalies by reconstructing time series and examination of error contextually, and anomaly score is a function of the output of both Generator and Discriminator. The method is unsupervised learning, meaning no prior knowledge about anomalies is provided during training. We have found this work interesting since No precise segmentation is possible, meaning they have focused on signals that cannot be segmented. Also, the length of the anomaly sequence is variable. Adversarial attacks on time series models are another related work. In [58], the authors propose adversarial attacks on deep learning time series classifiers using the fast gradient sign method (FGSM) [46], and the basic iterative method (BIM) [59], [60]. However, their methods are considered as black-box attacks since, in both approaches, adversarial samples are generated using the gradient of ResNet [61], rather than the targeted network. Lastly, [62] proposes adversarial attacks on Convolutional Neural Network (CNN), LSTM, and Gated Recurrent Unit (GRU) networks as multivariate time series regressions where the adversarial samples are crafted using FGSM and BIM.

4.2 Methodology

The robustness of neural point processes to natural shocks and adversarial attacks remains an open problem to date. We leverage existing research on adversarial attacks in vision and signal processing and extend such methods, when possible, to temporal point processes.

We additionally examine the transferability of attacks and compare their performance in transferred black-box settings. We especially believe methods developed for time series data will be applicable to point processes since they share common characteristics such as sequential and noisy data; the difference is that point processes model discrete events using a continuous intensity and time series bin events or other variables in time. For each data point x in our real dataset, we allow the perturbations to be within the l_∞ ball around x as has been proposed to be an acceptable notion for adversarial perturbations[46], [59].

4.2.1 Contribution

Despite the remarkable success of deep neural networks (DNNs), they suffer from serious vulnerabilities to adversarial attacks. Vulnerability's examination of DNN in computer vision and natural language processing has received attention recently. However, To the best of our knowledge, we are the first to explore the adversarial methods for point processes modeled by DNN and examine their performance. In particular, I) We propose several adversarial attack approaches to generate white-box and black-box adversarial perturbation against point processes modeled by deep learning. II) We show how adversarial attacks can disturb underlying parameters of point processes which are considered a threat to parametric modeling. Furthermore, III) We illustrate how susceptible deep point processes are to natural shocks and non-stationary changes in data.

4.2.2 Models

This work is limited to three type of point processes, non-stationary Poisson, self-correcting, and Hawkes processes, where the main focus is Hawkes processes. In non-stationary Poisson process, unlike regular Poisson process, the average rate of events is allowed to change by time. Non-stationary Poisson has all properties of a Poisson process, except for the fact that the intensity is a function of time, i.e. $\lambda = \lambda(t)$, instead of being fixed.

A point process N is called self-correcting if $cov(N(s,t), N(t,u)) < 0$ for $s < t < u$. In this formulation, *cov* denotes the co-variance of the two quantities [63]. Intuitively, due to the negative correlation, past points' occurrence, inhibits the future points' occurrence [64].

Lastly, in Hawkes processes [65]–[67], the event rate is not fixed, but is dependent on some random inputs, including the history of the process. Hawkes process is a self-exciting process, each arrival increases the rate of future arrivals for some time and is determined by a background Poisson process $\lambda_0(t)$, which reflects spontaneous events and at each event in the history a Poisson process g is centered at that event reflecting the increase in the intensity in near future. In summary, the intensity of the Hawkes process can be modeled as follows

$$\lambda(t) = \lambda_0(t) + \sum_{t_\mathrm{i}<t} g(t - t_\mathrm{i}), \tag{4.3}$$

where $\lambda(t)$ donates the event rate at time t [66].

In this work, we explore the performance of two deep neural networks-base point processes. For the exponential hazard (EXP) model as proposed by the authors of [43] and followed by [27], the inter-event time, $x_\mathrm{i} = (t_\mathrm{i} - t_{\mathrm{i}-1})$ is fed into a the RNN and the hidden unit of RNN is updated by $h_\mathrm{i} = f(W^h h_{\mathrm{i}-1} + W^x x_\mathrm{i} + b_h)$. Here f represents the activation function, and W^h, W^x, and b_h are the recurrent weight matrix, input weight matrix, and bias term, respectively [43]. The conditional intensity is a function of the elapsed time from the latest event and the hidden state of the RNN, as:

$$\lambda(t|H_t) = \phi(t - t_\mathrm{i}|\mathbf{h}_\mathrm{i}) \tag{4.4}$$

and ϕ is a non-negative function that is the hazard function with the following form as assumed by [43]:

$$\phi(\tau|h_\mathrm{i}) = \exp(w_t\tau + v^\phi.\mathbf{h}_\mathrm{i} + b^\phi) \tag{4.5}$$

And $\tau_\mathrm{i} = t_{\mathrm{i}+1} - t_\mathrm{i}$ is the inter-event interval.

The other examined model is the fully neural network-based (NN) model for general temporal point process [68], that relaxes the constraints on the time course for hazard functions of point processes while they are modeled using RNNs. In this model, the cumulative inten-

sity function is modeled by the integral of intensity function and the instantaneous intensity is obtained by taking the derivative of the cumulative intensity function. Such a model allows us to have flexible and general intensity function with exact evaluation. Formally, instead of modeling the hazard function ϕ, in this model the cumulative hazard function $\Phi(\tau, \mathbf{h}_\mathrm{i})$ is modeled where;

$$\Phi(\tau, \mathbf{h}_\mathrm{i}) = \int_0^\tau \phi(s, \mathbf{h}_\mathrm{i}) ds \tag{4.6}$$

And, one can achieve the hazard function by;

$$\phi(\tau, \mathbf{h}_\mathrm{i}) = \frac{\partial \Phi(\tau, \mathbf{h}_\mathrm{i})}{\partial \tau} \tag{4.7}$$

In this setting, $\mathbf{h}_\mathrm{i}$ is the hidden state of the RNN and $\tau_\mathrm{i} = t_{\mathrm{i}+1} - t_\mathrm{i}$ is the inter-event interval [68].

Although the main contribution of existing work is temporal point processes, it is worth mentioning that the proposed attacks are extendable to multivariate point processes. Moreover, in multivariate settings, other features such as event type are available that may make models more robust or fragile against attacks since different features are available to constrain with from both attacker and defender point of view. In next part, we describe employed adversarial attacks.

4.2.3 Adversarial attacks

In classification scenario, adversarial attacks against a model given input x and label y for the classifier $C(x)$ is expressed as an optimization [69]: $\underset{x^{adv}}{\text{argmin}} l(x^{adv}, x)$ s.t;

$$\begin{cases} C(x^{adv}) \neq C(x) & \text{untargeted attack} \\ C(x^{adv}) = y_t & \text{targeted attack, } y_t \text{ is the desired label} \end{cases}$$

where $l()$ is a distance metric between samples in input space (e.g., the l_2 norm).

In contrast and in regression problems, adversarial attacks can be defined based on numerical instability of the models. The numerical (in-)stability of an algorithm is defined

based on the extent to which a function's output changes with changes in the input [70] and adversarial attacks are toward increasing the instability of the model where defense mechanisms decrease it. Formally, considering a neural regression model $T(x, \theta)$, where $T : \mathbb{R}^{N_0} \rightarrow \mathbb{R}^{N_m}$ has N_0 scalar inputs and N_m scalar outputs, the objective of $T(x, \theta)$ for $x \in \mathbb{R}^{N_0}$ and corresponding target $y \in \mathbb{R}^{N_m}$ is [71]

$$\underset{\theta}{\text{argmin}} l(T(x, \theta), y) \tag{4.8}$$

Then given $T(x, \theta)$, in the adversarial attack setting, the objective of adversary with budget ϵ is to maximize the instability of $T(x, \theta)$, which is mathematically formulated below

$$\underset{||x_{adv} - x||_p \leq \epsilon}{\text{argmax}} \; l(T(x_{adv}, \theta), y) \tag{4.9}$$

where $p = 1, 2$, or ∞.

Notice that the optimal solution, x_{adv}, to the above optimization problem is not unique. In the current work and for the adversarial attacks, among all possible solutions, by performing a search on candidates for ϵ, we choose the minimum candidate that satisfies the adversary's objective.

Furthermore, depending on the algorithm utilized in adversarial attack generation and the degree of provided information to the attacker, attacks can be white-box or black-box attacks. For white-box attacks, the attacker is fully aware of the internals of the target model and its weights (θ) and uses the model's gradient to find the vulnerable regions of the input space that affects the model's output drastically. In a black-box setting, no information about the attacked network is provided to the attacker. However, the attacker can compose transferable adversarial perturbations to the target model using an alternate model. In existing work, we have crafted the adversarial samples by strategies presented in the following to attack temporal point processes.

Fast Gradient Sign Method (FGSM)

FGSM[2] [46] is a bounded attack initially proposed for the visionary. Here, we extend it to point process regression, and the adversarial sample is formed by using perturbations in the direction of the gradient. The adversarial sample generated by the FGSM can be formulated as

$$X^{adv} = clip(X + \psi(\theta, X), min(X), max(X)) \tag{4.10}$$

Where,

$$\psi(\theta, X) = \epsilon \cdot sign(\nabla_x \mathcal{J}(\theta, X)) \tag{4.11}$$

where $\mathcal{J}(\theta, X))$ is the required training's cost of the neural network, ϵ is the perturbation factor, and $clip(\cdot, a, b)$ function squeezes its input to the range of $[a, b]$. Such a transformation of input X increases the loss. Finally, for l_p norms with $p = 1, 2$, $\psi(\theta, X)$ is calculated as [72]

$$\psi(\theta, X) = \epsilon \cdot \frac{\nabla_x \mathcal{J}(\theta, X))}{||\nabla_x \mathcal{J}(\theta, X))||_p} \tag{4.12}$$

Iterative Fast Gradient Sign Method (iFGSM)

ifGSM, also known as Basic Iterative Method (BIM) [59], [60], is an iterative form of FGSM that, instead of one single step, k steps attack with a budget α is applied iteratively as presented in (4.13).

$$\begin{aligned} X^{adv0} &= X \\ X^{adv_{t+1}} &= X^{adv_t} + \alpha \cdot sign(\nabla_x \mathcal{J}(\theta, X^{adv_t})) \\ X^{adv_{t+1}} &= \mathbf{clip}(X^{adv_{t+1}}, X^{adv_{t+1}} - \epsilon, X^{adv_{t+1}} + \epsilon) \\ X^{adv} &= X^{adv_k} \end{aligned} \tag{4.13}$$

The iFGSM, as mentioned above, is based on l_∞.

[2]↑The term *sign* is referring to l_∞ norm used in this attack.

Projected Gradient Descent (PGD)

PGD attack as presented by [60] without random start, is a more potent iterative version of FGSM. However, unlike the original PGD, in this work, we apply PGD as a generalization of iFGSM with random initialization.

$$\begin{aligned} X^{adv0} &= X + \mathbf{clip}(\mathcal{N}(0^d, I^d), -\epsilon, +\epsilon) \\ X^{adv_{t+1}} &= X^{adv_t} + \alpha \cdot sign(\nabla_x \mathcal{J}(\theta, X^{adv_t})) \\ X^{adv_{t+1}} &= \mathbf{clip}(X^{adv_{t+1}}, X^{adv_{t+1}} - \epsilon, X^{adv_{t+1}} + \epsilon) \\ X^{adv} &= X^{adv_k} \end{aligned} \tag{4.14}$$

PGD is also considered as an iterative generalization of FGSM where the compact formulation can be written as in (4.15) to, where $\mathcal{P}$ is the projection onto the ball of interest,e.g., clipping for l_∞ norm.

$$\mathbf{repeat}\delta := \mathcal{P}(\delta + \alpha \nabla_\delta l(h_\theta(X + \delta), y)) \tag{4.15}$$

In this generalized formulation of PGD, h_θ is the logits of the networks for, l is the loss function and δ is the parameter that needs to be optimized in adversarial attack, and $\delta = \epsilon$ is the lower bound for the objective. We discuss the optimization in next chapter. For both iFGSM and PGD, we consider $k = 10$ and $\alpha = \epsilon/10$.

Momentum Iterative Fast Gradient Sign Method (miFGSM)

miFGSM [73] is a transformation of iFGSM such that, before applying FGSM with a budget α, the gradient of the previous t steps with a decay factor μ is employed to update the gradient at step t+1. In this approach, the update directions are stabilized, and the algorithm skips poor regional maxima during the iterations. Therefore, the crafted adversarial samples are more transferable. However, despite the high cost of miFGSM attack, as we see in Section

4.4, it is not generating more transferable samples in comparison to previous single step and iterative attacks.

$$
\begin{aligned}
X^{adv0} &= X, g_0 = 0 \\
g_{t+1} &= \mu \cdot g_t + \frac{\nabla_x \mathcal{J}(\theta, X^{adv_t})}{|\nabla_x \mathcal{J}(\theta, X^{adv_t})|_1} \\
X^{adv_{t+1}} &= X^{adv_t} + \alpha \cdot sign(g_{t+1}) \\
X^{adv_{t+1}} &= \textbf{clip}(X^{adv_{t+1}} X^{adv_{t+1}} - \epsilon, X^{adv_{t+1}} + \epsilon) \\
X^{adv} &= X^{adv_k}
\end{aligned}
\tag{4.16}
$$

RAND+FGSM (R+FGSM)

R+FGSM has been proposed by [74] to attack the adversarially trained neural networks. Here, in one step, a small random perturbation with size α is applied to the input before applying FGSM of $(\epsilon - \alpha)$ cost. R+FGSM is a randomized, single-step and computationally efficient form of PGD.

$$
\begin{aligned}
X^{adv} &= X + \alpha \cdot sign(\mathcal{N}(0^d, I^d)) \\
X^{adv} &= X^{adv} + (\epsilon - \alpha) \cdot sign(\nabla_x \mathcal{J}(\theta, X^{adv}))
\end{aligned}
\tag{4.17}
$$

Saliency Map (SM)

A saliency map in computer vision indicates the level of significance of a pixel to the human visual system that has application in region-of-interest extraction, image cropping, image captioning, and beyond [75], [76]. Similarly, we propose another single step, low cost adversarial attack where we first identify the *important* events, depending on the gradient of the neural network at a particular event, within the input sequence and then perturb

identified events by the adversarial saliency map approach to achieve the adversary sample. Formally, we propose the following attack strategy

$$\begin{aligned} X^{adv} &= X \\ X_{\mathrm{i}}^{adv} &= X_{\mathrm{i}} + \epsilon \cdot sign(\nabla_x \mathcal{J}(\theta, X_{\mathrm{i}})) \\ X_{\mathrm{j}}^{adv} &= X_{\mathrm{j}} - \epsilon \cdot sign(\nabla_x \mathcal{J}(\theta, X_{\mathrm{j}})) \end{aligned} \tag{4.18}$$

where event i maximizes $\nabla_x \mathcal{J}(\theta, X)$ and event j minimizes it w.r.t. the input event sequence. $\mathcal{J}(\theta, X)$ is the neural network's training cost, and ϵ is perturbation factor. In this approach, $2 \leq |a| \leq |X|$, where a is the set of adversarial components in the input sequence X and $|\cdot|$ is the notation for the number of components in a set.

As aforementioned, our SM attack is a low cost, w.r.t. the number of iterations, l_∞ attack. However, in [76], an iterative l_0 attack has been proposed based on the saliency map for image classification in which the adversarial attack modifies the input image's elements until the misclassification goal is acquired or the total number of modified elements surpasses the attack's budget. Compared to single-step adversarial attacks, iterative approaches result in more effective attacks. Nevertheless, the cost of crafting such an attack is relatively higher than single-step attacks.

Gradient descent (GD)

In gradient descent approach, we remove the first event from the input sequence and add a new event time within interval of first and last event of the sequence using gradient descent. The procedure of gradient decent attack is presented in Algorithm 1.

Algorithm 1 Gradient descent adversarial attack procedure

Parameter: Original event sequence (X), Perturbation factor ϵ, Perturbation step α

Output: Adversarial event sequence (X^{adv})

procedure GDAdversarial(X, ϵ, α)
    $X^{adv} \leftarrow X$ ▷ Initialization
    $event \leftarrow \text{genRandomVal}[\min(X^{adv}), \max(X^{adv})]$
    $idx \leftarrow \text{genRandomInt}[0, length(X^{adv})]$
    $X^{adv}.pop(0)$
    $X^{adv}.\text{insert}(idx, event)$
    while $(\nabla_x \mathcal{J}(\theta, X^{adv}_{idx})$ is changing) and $\alpha < \epsilon$ **do**
        $event \leftarrow event + (\alpha \cdot \nabla_x \mathcal{J}(\theta, X^{adv}_{idx}))$
        $X^{adv}[idx] \leftarrow event$
        $idx \leftarrow getIdx(X^{adv}, event)$
    end while
    return X^{adv}
end procedure

Time manipulation (TM)

In point process applications, e.g., crime forecasting based on reported crimes to the police departments, one can easily report a fake crime, misleading the point process algorithm. Following the same context, we propose a time manipulation attack. Here, we remove the first event in the event sequence and add a random event within the interval of the first and last event's time regardless of the model weight and architecture. Therefore, we consider TM a black-box version of the GD attack since it is independent of the network architecture and cost of training. Additionally, TM is the cheapest attack since it is a single-step attack independent of the network's gradient.

Transferability and transferable black-box setting

Regardless of the strength of adversarial attacks against deep learning models, recently, the transferability of adversarial samples has raised concerns in literature[46], [77], [78]. In this phenomenon, adversarial samples between two independent trained models are transferred. In the current work, we study this phenomenon by proposing transferable adversarial attacks, such that we craft the adversarial instances by the white-box attacks, FGSM, iFGSM, PGD, miFGSM, R+FGSM, SM, and GD presented in Section 4.2.3, on neural networks with more trainable parameters. Then, we used the crafted adversarial samples to attack small networks with fewer trainable parameters. To control the parameters, attackers can have either more neural network layers, RNN units, or hidden units. We present details in Table 4.1. Specifically,

- RNN units: Means the number of units in an RNN layer [68].
- Network depth (layers): The number of hidden layers of the cumulative hazard function network.
- Hidden neural network (NN) units: The number of units in each hidden layer of the fully neural network model [68].

Table 4.1. Hyper-parameters of transferability check for white-box adversarial attacks. An example of an attacker has either 4 layers, 256 hidden units, or 256 RNN units, and FGSM adversarial approach.

	Attacked	Attacker	Approach
Network depth	2	4	miFGSM, R+FGSM
Hidden units	64	256	PGD, iFGSM
RNN units	64	256	FGSM, SM GD

4.2.4 Baseline and Metrics

Baseline corresponds to the standard case when a model is built without any adversarial assumptions. We will compare each potential adversarial attack's performance when under

attack to the baseline and its performance when no attack has occurred. Our metrics in this comparison are:

- **Mean Negative Log Likelihood** Our first metric in the mean negative log likelihood which is also part of the loss function of neural network models in this work. Since NLL depends on the predictive uncertainty, it's a reasonable metric in evaluating predictive uncertainty [79], [80].

- **Mean Absolute Error**

$$MAE = \frac{1}{K}\sum_{k=1}^{K} ||T(x_k) - y_k||_1 \tag{4.19}$$

$$MAE_{adv} = \frac{1}{K}\sum_{k=1}^{K} ||T(x_k^{adv}) - y_k||_1 \tag{4.20}$$

- **Fooling Error**

$$FE = \frac{1}{K}\sum_{k=1}^{K} ||T(x_k^{adv}) - T(x_k)||_q \tag{4.21}$$

- **Symmetric Mean Accuracy Percentage Error [71]**

$$SMAPE = \frac{2}{K_+}\sum_{k=1}^{K_+} \frac{||T(x_k^{adv}) - y_k||_q - ||T(x_k) - y_k||_q}{||T(x_k^{adv}) - y_k||_q + ||T(x_k) - y_k||_q} \tag{4.22}$$

Where q norm in FE and SMAPE metrics, must match the l_p norm employed in generating adversarial attacks and SMAPE is limited to the K_+ positive elements in the summation.

Goodness of fit

In addition to previous metrics, and due to the nature of our problem and data, we are assessing the goodness of fit. The time-rescaling theorem and thinned residuals are two ways as following to ensure the deep learning models fit our data properly[81], [82];

- **Time-rescaling (Theorem)** [83]–[85] Consider $t_1, t_2, , t_k$ as realizations of a particular point process wherein its conditional intensity function is $\lambda^*(\cdot)$ over time $[0, T]$. If

$\lambda^*(\cdot) > 0$ over $[0, T]$ and $\Lambda(T) < \infty$, then $\{\Lambda(t_1), \Lambda(t_2), \ldots, \Lambda(t_k)\}$ as transformed points, form a unit rate Poisson process.

The time-rescaling is well-known in point processes and assessing the goodness of fit. In its univariate form, by using this theory, "any" point process with an integrable conditional intensity function can be transformed into a unit rate Poisson process [83]. Therefore, if a deep learning model has successfully modeled a point process, the the inter-event intervals of the orderly point process should be transformed, or rescaled, such that the result is a unit rate Poisson process.

- **Ordinary thinned residuals** [86]: Thinning data based on estimated λ matches Poisson process with rate m. Here, data point t_{i}, will be removed with probability $\frac{1-m}{\lambda(t_{\mathrm{i}}|H)}$. The remaining points assemble a Poisson process with $\lambda = m$ over the original domain $\mathcal{S}$.

  Where $\lambda(\cdot)$ is the conditional intensity function, $0 < m < max(\lambda(t_{\mathrm{i}}|H_{\mathrm{i}}))$ and $\mathrm{i} = 1, \ldots n$.

In Appendix A we report the result of the experiments that shows both models are capable of modeling the data properly.

4.3 Data

Similar to [43], [68], experiments are run on the following point process datasets. We split each dataset into train and test sets. The train set is then used to estimate the model's parameters in the training phase, and the model's performance on the test set is quantified by the evaluation metrics.

4.3.1 Hawkes Process

In this work, we have simulated [66] 100,000 event times from the Hawkes process with the sum of exponentials kernel, and the conditional intensity function is given by:

$$\lambda(t|H_t) = \mu + \sum_{\mathrm{j}}^{M} \sum_{t_{\mathrm{i}}<t} \alpha_{\mathrm{j}} \beta_{\mathrm{j}} \exp\{\beta_{\mathrm{j}}(t - t_{\mathrm{i}})\} \tag{4.23}$$

Where $M = 1$, $\mu = 0.2$, $\alpha_1 = 0.8$, and $\beta_1 = 1.0$. In (4.23), μ is the background intensity, α is adjacency and represents the jump in intensity after the arrival of a new event, and β is the intensity decay rate after arrival.

In the exponentially decaying Hawkes process, $\frac{\alpha}{\beta}$ is known as *branching ratio* The ratio $\frac{\alpha}{\beta}$ is known as *branching ratio* and is the declaration of the Hawkes process regime and if the Hawkes process explodes (super-critical regime and $\frac{\alpha}{\beta} > 1$), or not [66]. The Hawkes process can be in three regimes depending on the branching ratio. Specifically, if the branching ratio is 0, the model will become a non-homogeneous Poisson process since the initial immigrants will trigger no further events. Indeed, this is Hawkes process as a generalization of the Poisson process that considers the time and history of a process [87].

For $\frac{\alpha}{\beta}$ ratio, the critical threshold is 1, which splits the process into subcritical ($\frac{\alpha}{\beta} < 1$) and supercritical ($\frac{\alpha}{\beta} > 1$) regimes. If $\frac{\alpha}{\beta} > 1$ for a sustained amount of time, the modeled intensity may explode.

Finally, if the background intensity is a constant and the process is in the subcritical regime ($\frac{\alpha}{\beta} < 1$), the branching ratio can be seen as a probability [88].

4.3.2 Self-correcting Process

For Self-correcting process, 100,000 event times are simulated by the conditional intensity function that is given as $\lambda(t|H_t) = \exp(t - \sum_{t_i<t} 1)$.

4.3.3 Non-stationary Poisson Process (N-Poisson)

Finally for non-stationary Poisson dataset, 100,000 events time are simulated from the following conditional intensity function $\lambda(t|H_t) = 0.99 \sin(\frac{2\pi t}{20000}) + 1$ as suggested by [68].

4.3.4 Crimes and Covid-19

To assess the performance of models concerning non-stationary changes, such as the effect of the Covid-19 pandemic on reported crimes in Chicago, we use the reported crimes to the

Chicago police department from 2016 to 2018 as training, and crimes in 2019 and 2020 as test set, separately[3].

4.4 Results and Discussion

This section presents the results of our adversarial attacks on the deep point process models on the specified datasets. We present the predictive performance-related experiments on all point process datasets.

We see that the choice of metric is a critical factor in evaluating adversarial attacks' effectiveness. Specifically, we illustrate that MAE is the least expressive evaluation metric in an adversarial setting. Furthermore, through extensive experiments, we see our proposed PGD can be considered as a "universal first-order attack" such that by first-order, we mean the adversarial attacks solely depend on the gradient of the neural network as suggested by [59]. However, our results reveal that such a generalization is limited to first-order adversarial attacks utilizing the entire gradient vector. For single-point adversarial attacks, e.g., gradient descent method (GD), PGD can not be considered a universal attack.

From the results, iFGSM has the best overall performance in white-box attacks against both fully neural network (NN) and exponential kernel (EXP) models. Furthermore, similar to [59], our results suggest that the transferability of attacks decreases as the power of attack increases, and single-step attacks have more transferability in comparison to iterative attacks.

We discuss adversarial attacks manipulating the parameters and branching ratio of the Hawkes process. Finally, we investigate the effect of non-stationary abrupt changes on the models' performance, using the crimes dataset during the Covid-19 era.

It is essential to emphasize the role of l_p norm in adversarial attacks. In generating adversarial samples, ϵ, the perturbation factor, in l_2 norm perturbations is more significant than l_∞ perturbations since the volume of l_2 ball in n-dimensional space with radius r is smaller than the volume of l_∞ ball in n dimension space with the same r. It is noteworthy to mention that ϵ is one of the constraints of the cost of adversarial attacks. Thus from the

[3]↑https://data.cityofchicago.org/Public-Safety/Crimes-Map

defense point of view as we discuss in next chapter, in adversarial training, l_∞ can increase the cost of adversarial attacks for the adversary.

Another point on the use of l_p norm is the characteristics of each norm for $p = 1, 2$, and ∞, which can be adapted to the context of data. In l_∞ adversarial attacks, the perturbation is allowed under the l_∞ ball, meaning l_∞ attacks make slight noise everywhere in the input sample. On the other hand, l_2 attacks lead to more localized perturbations in the input sample since we can trade-off a more significant perturbation in one point of the space for minor perturbation in another. Finally, under l_1 norm-based attacks, the sparsity in the perturbation is encouraged, which means only a few adjusted input sample elements is present in the input sample. In practice, the adversarial attacks under the l_1 norm are computationally expensive to craft. In all adversarial attacks in this chapter, we use l_∞.

In Figure 4.1, we present two adversarial samples along with the conditional intensity values for each sequence and simulation of the model prediction for the real and adversarial input. From the examples, it can be inferred that unlike their intensity, the adversarial samples are close to real sample in time space.

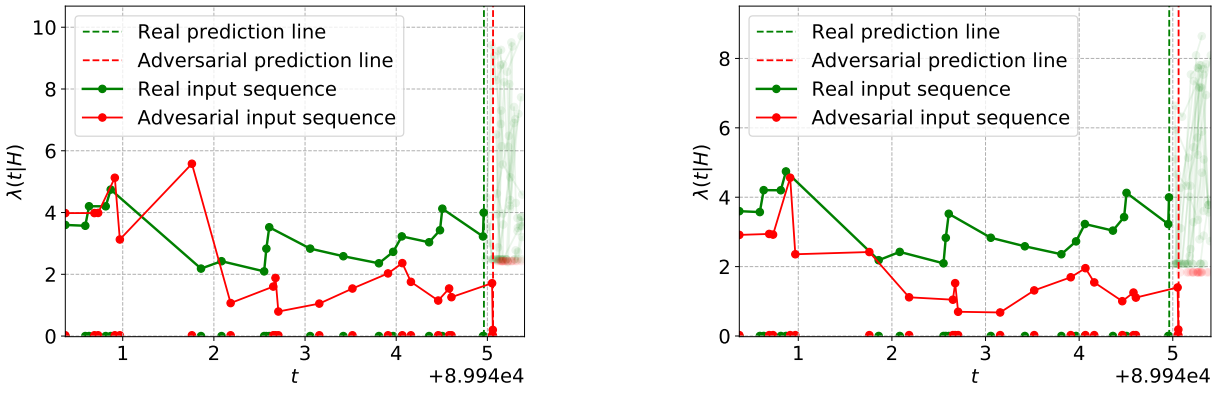


Figure 4.1. Real input sequence (Green) and the corresponded adversarial sequence (Red) on Hawkes data. Left: Transferred FGSM from larger network, layers = 4. Right: Transferred FGSM from larger network, RNN=256. The x-axis represents time and y-axis represents the conditional intensity.

In adversarial attacks against regression models, we expect some change in response for any change in the input. In this situation, the adversary aims for a dramatic change in output for a small change in input[70]. According to Figure 4.2, in white-box setting and to

have high transferability, the perturbation factor (ϵ) needs to be large enough, although with $\epsilon = 0.1$, the change in performance is statistically significant yet. We present the associate cost with each adversarial attack in Table 4.2. In all of our attacks, ϵ is set to 0.1 and $\alpha = \frac{1}{10}\epsilon$.

Table 4.2. Associated cost with white-box and black-box adversarial attacks on Fully neural network (NN) model and Exponential hazard (EXP) model

Attack	**Step perturbation**	**Perturbation factor**	**# of event^{adv}**	**Iterations**
PGD	α	ϵ	20	10
iFGSM	α	ϵ	20	10
R+FGSM	NA	ϵ	20	1
miFGSM	α	ϵ	20	10
FGSM	NA	ϵ	20	1
SM	NA	ϵ	2	1
GD	α	$\leq \epsilon$	1	≤ 100
TM	NA	ϵ	1	1

# of event^{adv} is the number of adversarial events in each input sequence of events.
Each iteration requires $\nabla_x \mathcal{J}(\theta, X)$ calculation

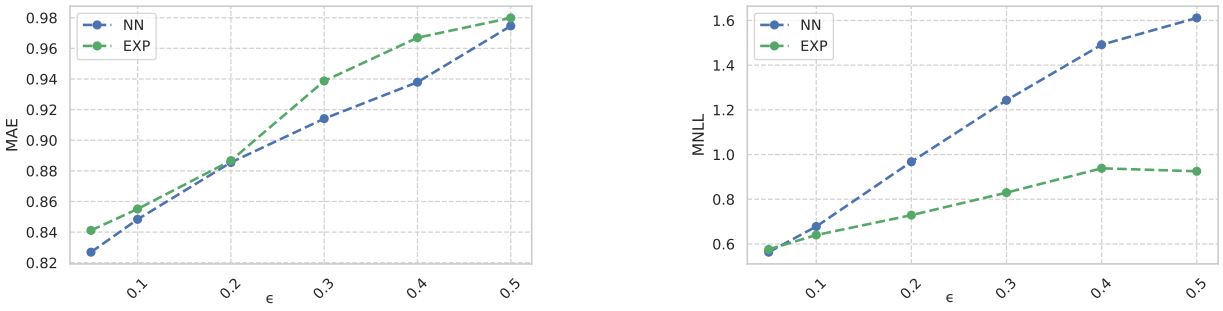


Figure 4.2. FGSM attack's sensitivity to perturbation factor. Left: MAE, Right: MNLL. Fully neural network (NN) is in blue and exponential hazard (EXP) in green.

In Table 4.3, the performance of the fully neural network model on the specified datasets as well as the most effective white-box and black-box attacks are presented. Consecutively, in Table 4.4, we present the performance of exponential hazard model on the same datasets and the most effective attack mechanisms.

Table 4.3. Predictive performance of fully neural network (NN) model against the most effective* white-box and black-box attacks.

Point process	**Attack**	**MNLL** (e-01)	**MAE** (e-01)	**FE** (e-02)	**SMAPE** (e-01)
Hawkes	No Attack	4.1	7.77	NA	NA
	iFGSM	6.54	8.21	4.75	3.66
	TM	5.98	8.36	2.12	7.08
NS-Poisson	No Attack	9.83	10.0	NA	NA
	iFGSM	10.41	10.38	4.64	1.573
	TM	12.91	11.46	40.56	4.05
Self-correcting	No Attack	8.21	4.97	NA	NA
	iFGSM	16.31	9.52	48.00	7.77
	TM	11.94	7.05	49.79	8.32

∗ We recognize a white-box attack as "effective", if both FE and SMAPE are the highest among first-order white-box attacks.

The column-wise scale of each metric is presented in () below it.

Table 4.4. Predictive performance of exponential hazard (EXP) model against the most effective* white-box and black-box attacks.

Point process	**Attack**	**MNLL** (e-01)	**MAE** (e-01)	**FE** (e-02)	**SMAPE** (e-01)
Hawkes	No Attack	4.63	8.01	NA	NA
	iFGSM	5.67	8.26	3.66	2.32
	TM	6.67	8.72	28.58	6.27
NS-Poisson	No Attack	9.7	10.2	NA	NA
	miFGSM	10.32	10.60	5.27	1.65
	TM	13.36	11.42	37.74	4.25
Self-correcting	No Attack	7.82	4.94	NA	NA
	iFGSM	18.86	9.65	50.41	8.01
	TM	11.49	6.91	49.02	8.36

∗ We recognize a white-box attack as "effective", if both FE and SMAPE are the highest among first-order white-box attacks.

The column-wise scale of each metric is presented in () below it.

As aforementioned, MNLL is a metric to quantify the predictive uncertainty. From Table 4.3 and Table 4.4, for instance, iFGSM and TM are both increasing the MNLL of the NN model. However, against the EXP model, iFGSM is not as effective as the NN model, but, it still increases the predictive uncertainty. In Table A.1 and Table A.2 in Appendix A, we provide the effect of all adversarial attacks using the evaluation metrics. Additionally, In Figure 4.3, we compare all attacks for both the NN and the EXP models. As it can be

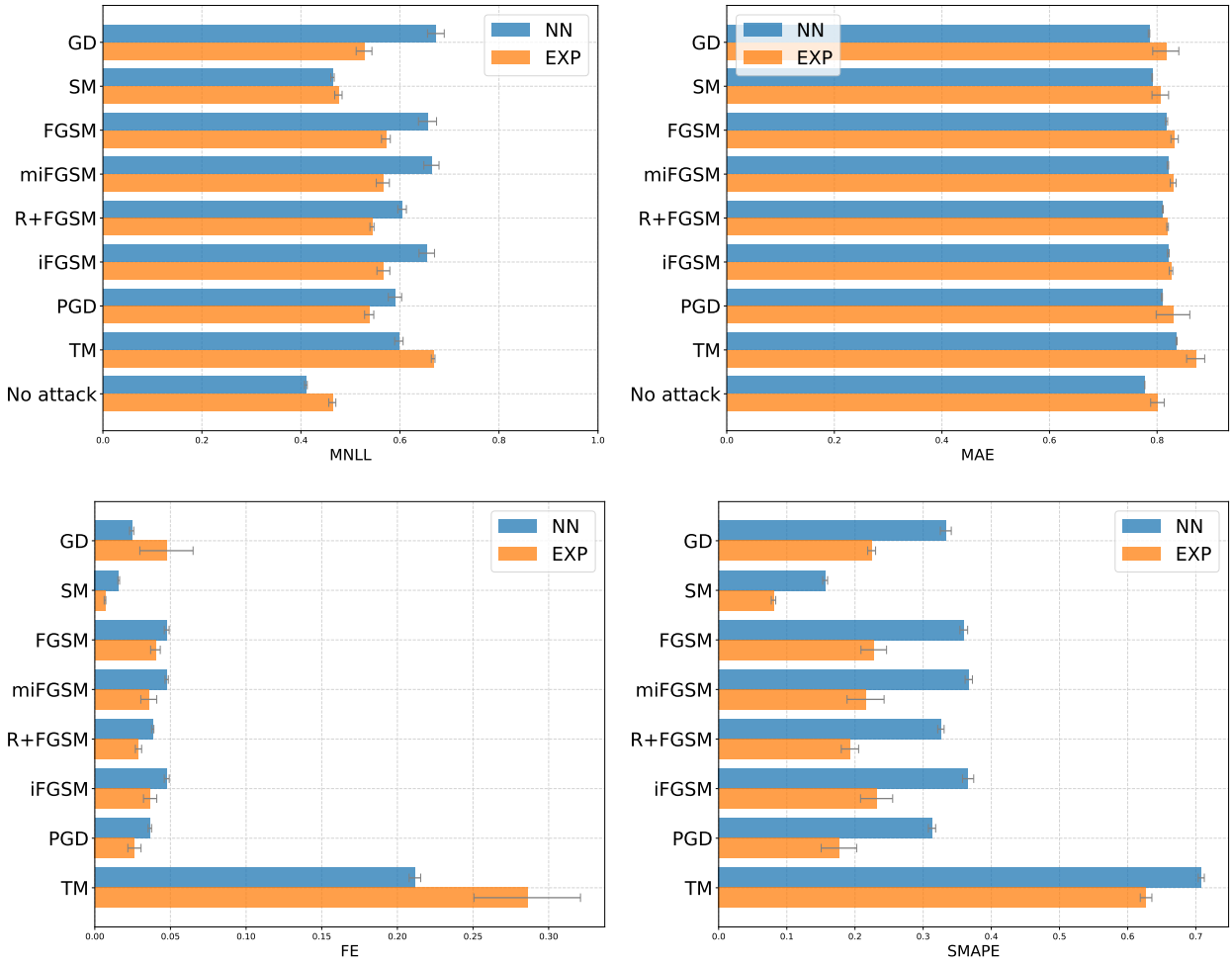


Figure 4.3. Effect of adversarial attacks against NN (Blue) and EXP (Orange) models on their predictive performance in terms of MNLL 4.2.4, MAE 4.2.4, FE 4.2.4, and SMAPE 4.2.4

inferred from Figure 4.3, MAE is the least expressive metric for comparing the performance of adversarial attacks and the model's robustness. One reason for that is the error quantification

in MAE that considers the "absolute value of error" and therefore MAE is not sensitive to adversarial attacks. Yet, an adversarial attack can affect the input sample while keeping the absolute error low, and again, choice of metric is critical in evaluating adversarial attacks.

In Figure 4.4, we compare the effectiveness of adversarial attacks against the NN model in a sorted presentation. As illustrated, with respect to Fooling error (FE) and Symmetric Mean Accuracy Percentage Error (SMAPE), TM is the most effective attack, although, unlike others, TM is a single-step black-box attack, and the perturbation is local. Conversely, in first-order attacks, the perturbation is dispersed over the input sample, and the attack affects every event time in the sequence utilizing the gradient of the network. Such outcomes imply despite the history term in the Hawkes process's conditional intensity, the model is vulnerable to spikes in the input and more robust against global perturbations.

In adversarial attacks against point processes, the neural point process model's predictive performance is the goal of the attacks. Not surprisingly, the estimated parameters are also vulnerable if the neural point process model is used to estimate the intensity function. Considering an application of the Hawkes process, e.g., anomaly detection, the intensity of the event's arrival is increased by an event occurrence and then exponentially decays towards the baseline level. Therefore, the values of μ, α, and β influence the model's behavior in predicting the arrival time of the event.

In the studied Hawkes process in the current work, since the background intensity, μ, is fixed, depending on the branching ratio, the Hawkes process will explode and will be in super-critical regime, if $\frac{\alpha}{\beta} > 1$, as we discussed in Section 4.3.

From Table 4.5, the test set is a subcritical ($\frac{\alpha}{\beta} < 1$) Hawkes process. Moreover, on adversarial sets, for both the NN and EXP models, all adversarial sets, except GD against NN, are subcritical Hawkes processes. However, the predicted events by the NN model are all super-critical Hawkes processes, except for predictions in presence of SM and GD attacks. On the other hand, the predictions made by the EXP model are all subcritical Hawkes processes, except for GD. Preserving the Hawkes process's regime is an important characteristic and results suggest the NN model cannot maintain it. From here, we recommend considering the effects of adversarial attacks on the predictive performance and point process parameters to be regarded as *Micro* and *Macro* effects, respectively. In the Micro effect, the effect

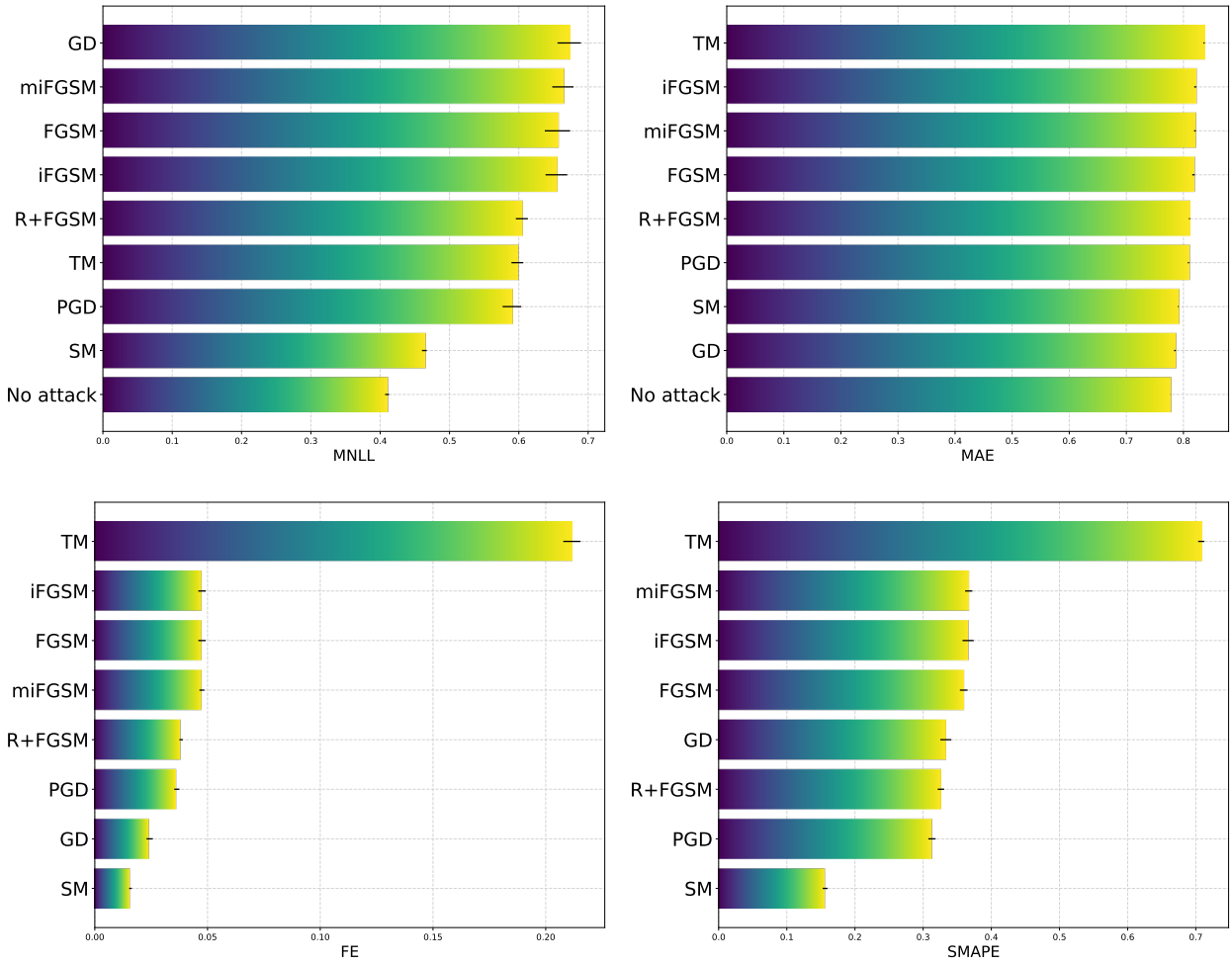


Figure 4.4. Effect of each attack on predictive performance of NN model in terms of MNLL (top left), MAE (top right), FE (bottom left), and SMAPE (bottom right).

Table 4.5. Adversarial attack's Macro effect on fully neural network (NN) and exponential hazard (EXP) models w.r.t. Hawkes process parameters, μ, α, and β.

	Fully neural network			**Exponential hazard**		
	$\boldsymbol{\mu}$ (e-02)	$\boldsymbol{\alpha}$ (e-01)	$\boldsymbol{\beta}$ (e-01)	$\boldsymbol{\mu}$ (e-02)	$\boldsymbol{\alpha}$ (e-01)	$\boldsymbol{\beta}$ (e-01)
Test set	1.29	8.74	9.11	1.29	8.74	9.11
First-order attacks	99.8	10.0	13.0	99.8	10.0	13.0
GD	99.6	9.9	9.08	99.6	9.9	10.2
TM	99.6	9.99	11.5	99.6	9.99	11.0
Prediction in presence of adversarial attacks						
No Attack	1.25	8.79	8.43	1.32	8.70	9.43
PGD	1.20	8.82	7.94	1.32	8.71	9.19
iFGSM	1.17	8.84	7.77	1.32	8.71	9.16
$R + FGSM$	1.18	8.83	7.84	1.32	8.72	9.11
miFGSM	1.18	8.84	7.80	1.32	8.71	9.15
FGSM	1.18	8.83	7.83	1.32	8.71	9.24
SM	1.23	8.81	8.19	1.32	8.70	9.35
GD	1.12	8.88	7.372	1.23	8.81	8.20
TM	2.73	7.30	53.5	1.54	8.41	10.00

The column-wise scale of each parameter is presented in () below it.

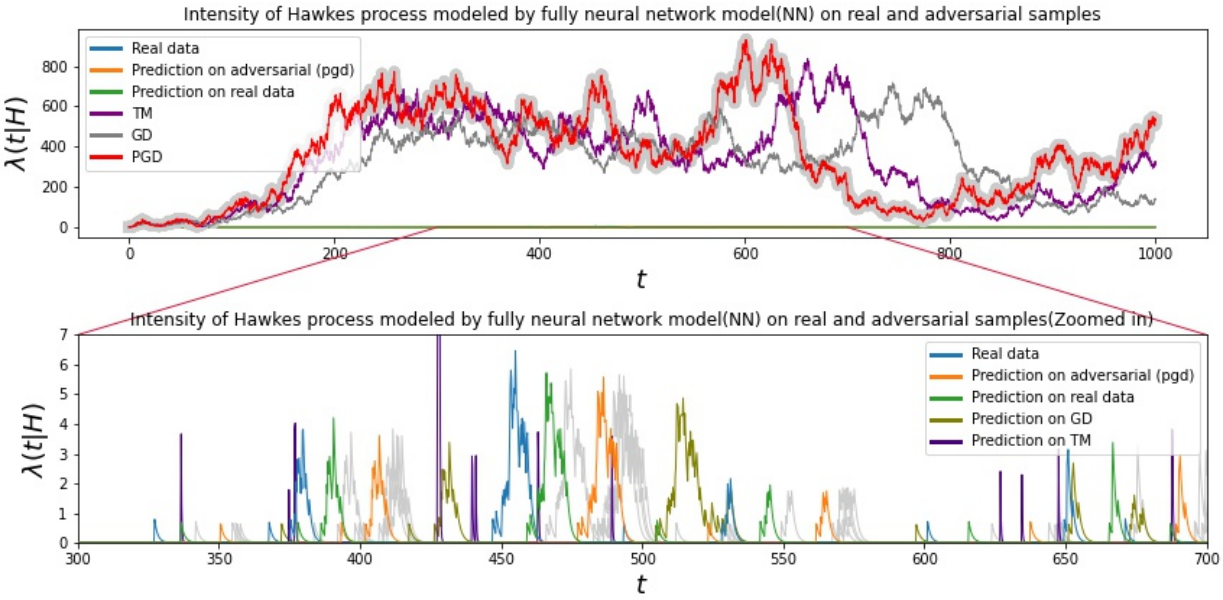


Figure 4.5. Adversarial attack's macro effect on the fully neural model (NN) parametric modeling performance. Top: Conditional intensity of real events (Blue), predicted events (Green), and adversarial events generated by PGD attack (Red), GD (Gray), and TM (Purple), respectively. Bottom: Enlarged view. Note that adversarial samples' intensity, $\lambda(t|H)$, range is far from the real and predicted samples. Here, $\lambda(t|H)$ of all other first-order attacks and corresponded predictions have been presented in "light gray" as they are tight to PGD. Additionally, we represent $\lambda(t|H)$ of GD and predicted event arrivals in the presence of GD in "Olive". The same for TM attack is shown in "Indigo".

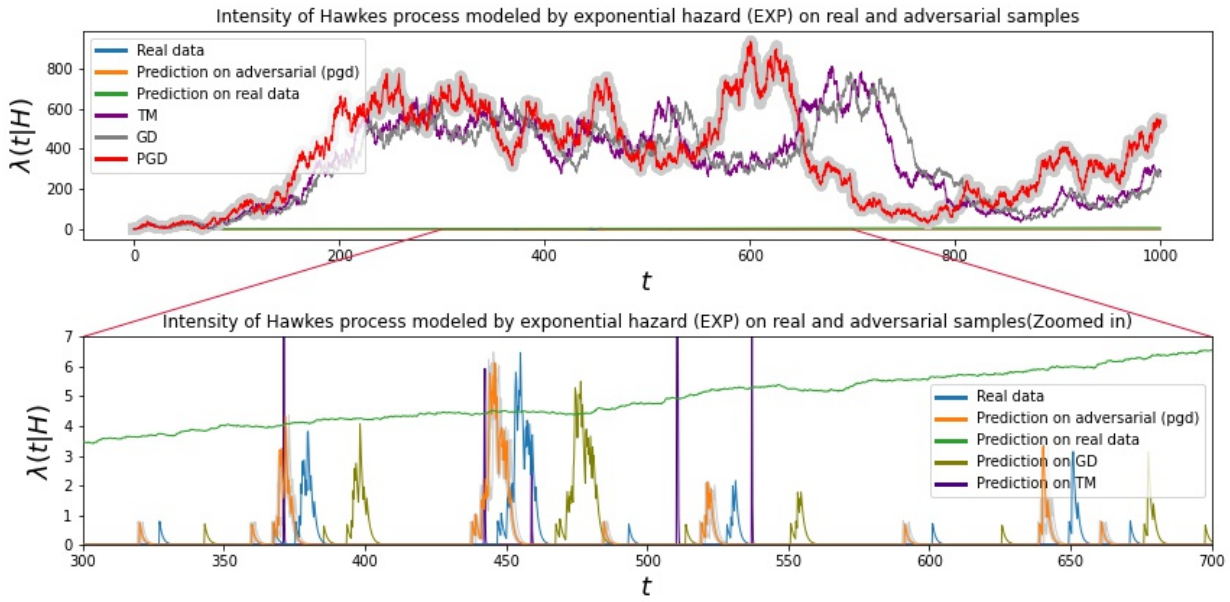


Figure 4.6. Adversarial attack's macro effect on the exponential hazard (EXP) parametric modeling performance. Top: Conditional intensity of real events (Blue), predicted events (Green), and adversarial events generated by PGD attack (Red), GD(Gray), and TM (Purple), respectively. Bottom: Enlarged view. Note that adversarial samples' intensity, $\lambda(t|H)$, range is far from the real and predicted samples. Here, $\lambda(t|H)$ of all other first-order attacks and corresponded predictions have been presented in "light gray" as they are tight to PGD. Additionally, we represent $\lambda(t|H)$ of GD and predicted event arrivals in the presence of on GD in "Olive". The same for TM attack is shown in "Indigo".

of an adversarial attack is limited to each input sequence and it's reflected in predictive performance. In contrast, for the Macro effect, the adversarial attack may affect the modeled point process parameters and this affect the parametric modeling performance. Presented results both on predictive performance and Hawkes process parameters suggest that although adversarial attacks show disagreement in their Micro effects, but all first-order white-box attacks agree on their Macro effects.

In Figures 4.5 and 4.6 we represent the effect of adversarial attacks on Hawkes parameters. As aforementioned and presented in Figure 4.5 and Figure 4.6, all first-order white-box attacks (light gray area around PGD), both against the NN model and the EXP model, share the underlying parameters, μ, α, and β and agree on their Macro effects. Additionally, although all adversarial sets have a greater intensity range in comparison to real data, but both models are predicting points (events) based on the adversarial input which the range of the predictions are between 0 and 10. Such a result suggest both models are successful in maintaining the range of intensity functions. However, if the adversarial sets, e.g., PGD, GD, etc. be fed to a parametric model, the model will failed in modeling them.

Moreover, in NN model, although first-order white-box attacks share their parameter, but there is a shift in intensity of predictions made by the model. However, in EXP model, the prediction on adversarial sets are also sharing their Hawkes's parameters. Considering single sample attacks such as GD and TM, they are not following the rest of adversarial attacks, but are more proximate to each other. The algorithm responsible for generating GD samples can explain the shift in parameters and intensity functions of TM and GD since the GD attack can be considered as a steady version of TM.

To evaluate the transferability of white-box attacks against NN and EXP models, for each attack, we perform attacks against independently trained networks with more complexity and larger architecture and then deploy the adversarial set to our original targeted models. To increase the complexity, we try hyper-parameters as discussed in the transferability Section 4. For the exponential hazard model, the "RNN units" is the only available hyper-parameter. In Figure 4.7 and Figure 4.8, we illustrate the transferability power of each attack against our models. The full version of results is available in Table A.3, A.4, A.5, A.6 in Appendix A. Our transferability investigation implies the susceptibility of both models to iFGSM

adversarial attack, the strongest attack w.r.t. predictive performance, is not increased in transferring. For the EXP model to have an single-step effective attack, the adversarial samples require to be crafted in a white-box setting. On the other hand, for the NN model, the adversarial objective is achievable in both black-box (transferred) and white-box settings.

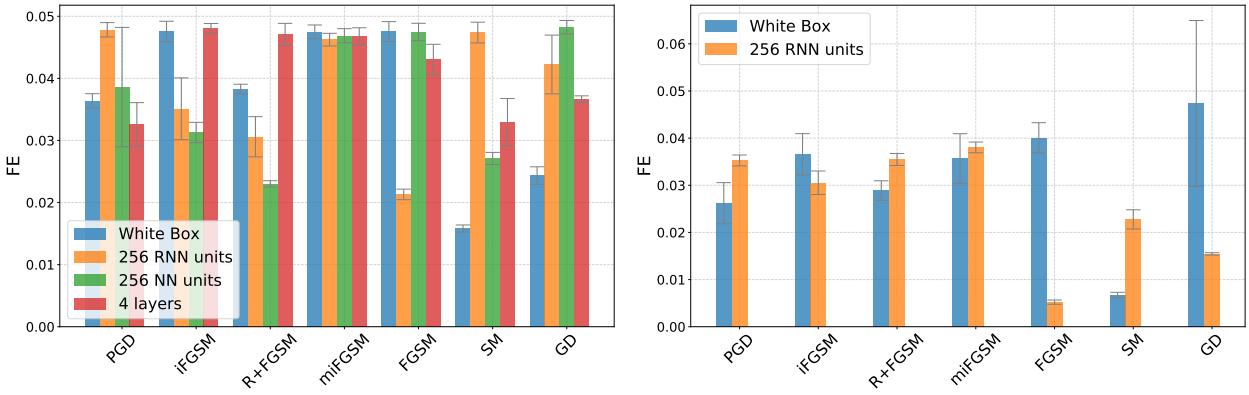


Figure 4.7. Fooling error (FE) of each attack in the white box and semi-black-box settings against (left) fully neural model (NN) and (right) exponential kernel hazard model (EXP). The attacker has either 4 layers, 256 neural units, or 256 RNN units for the semi-black-box setting.

Finally, in Table 4.6, we evaluate the performance of models with respect to non-stationary abrupt changes, Covid-19 pandemic's impact on reported crimes to Chicago Police Department in 2020. According to the results, fully neural network model has better performance in modeling crimes, but the exponential hazard model is more robust (% of change in MNLL) against non-stationary abrupt changes, although it's still sensitive to non-stationary changes.

Table 4.6. The performance of the models on crime prediction in 2019 and 2020. fully neural network hazard function (Left half), and exponential hazard model (Right half)

	Fully neural network		**Exponential hazard**	
Data	**MNLL** (e+00)	**MAE** (e+01)	**MNLL** (e+00)	**MAE** (e+01)
Crime(2019)	5.08	9.66	5.672	11.06
Crime(2020)	5.44	13.1	5.929	14.62

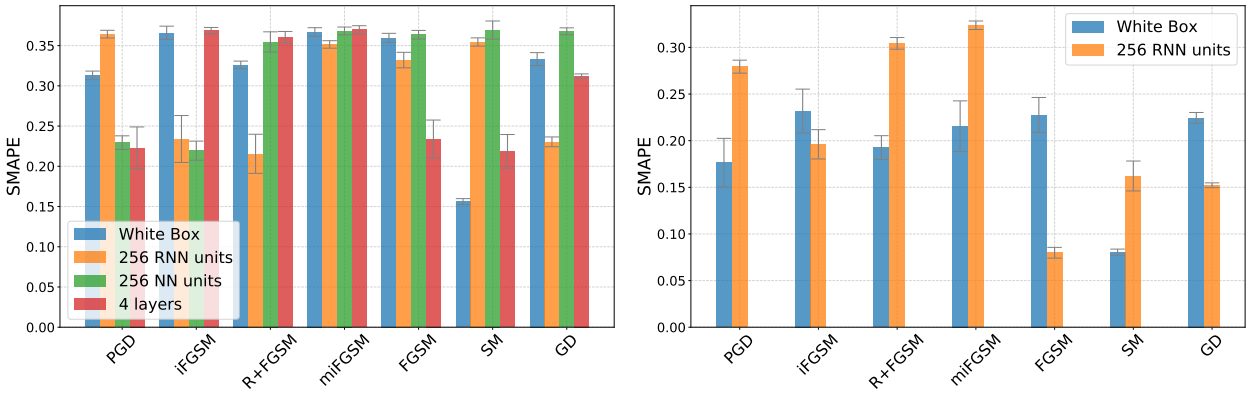


Figure 4.8. Symmetric Mean Accuracy Percentage Error (SMAPE) of each attack in the white box and semi-black-box settings against (left) fully neural model (NN) and (right) exponential kernel hazard model (EXP). The attacker has either 4 layers, 256 neural units, or 256 RNN units for the semi-black-box setting.

5. ADVERSARIAL ROBUSTNESS

5.1 Related works

From the optimization standpoint of adversarial robustness, the decision-making process of a machine learning model can be viewed as an empirical risk assessment, and the objective is to minimize the risk of decision-making by machine learning algorithms. Accordingly, in the presence of adversarial samples and adversarial training as a defense mechanism, the objective evolves to reduce the empirical adversarial risk, which is the objective of adversarial training. Adversarial training is a *MinMax* between adversary and model's loss function, so it can be written as

$$\min_{\theta} \frac{1}{\mathcal{D}} \sum_{x,y \in \mathcal{D}} \max_{||\delta|| \leq \epsilon} \mathcal{L}(\theta, (x+\delta), y) \tag{5.1}$$

where $\mathcal{D}$ is the clean dataset containing (x, y) pairs and $\mathcal{L}$ is the loss function.

In this optimization problem, the objective of the inner maximization problem is to find an adversarial sample of a given data point x that maximizes the loss function. At the same time, for the outer optimization, the goal is to find a set of parameters, θ, such that given the inner attack problem, the adversarial loss is minimized. Where the former statement is the precise definition of adversarial attacks against a network, and the latter is the problem of training a robust network using adversarial training techniques [60].

For a deep learning model, there are three strategies to solve the inner optimization problem; lower bound, exact solution, and upper bound. In the following, we discuss each strategy briefly.

Finding the lower bound for the inner optimization is equivalent to finding an adversarial attack since any possible δ will provide a lower bound resulting from a practical solution for the optimization problem that is also a local optimum. In practice, lower bound solutions are not always solid adversarial attacks, and therefore they are not utilized in adversarial training to provide robust models. However, they are still valuable for improving the robustness of the model.

Finding the exact solution for the inner optimization for some activation functions is achievable, which can be formulated as a combinatorial optimization problem. The combi-

natorial optimization then is solved through techniques such as mixed-integer programming [89], [90]. However, optimization via mixed-integer programming has substantial challenges in scaling to large models, and not all can be written as mixed-integer linear programs.

Ultimately, the optimization objective can be upper bounded using the network structure's relaxation strategy. In this approach, the relaxed version contains the original network while it is much easier to optimize exactly over, which involves building convex relaxations of the network structure itself [91], [92]. The noteworthy point is that since this method is based on the relaxed version, it does not produce an actual adversarial example for the original model. Nevertheless, it can certify that a network is provably robust against an adversarial attack, and if being utilized in adversarial training, the result is a provably robust model.

All considered, the common Fast Gradient Sign Method (FGSM) which is a single step l_∞ bounded adversarial attack, we discuss in 4.2.3, is lower bounding the inner maximization that is maximizing the loss function given the optimal δ which is

$$\delta^* = \epsilon \cdot sign(\nabla_\delta \mathcal{L}(\theta, (x+\delta), y)$$

A more powerful attack, such as PGD adversarial attack 4.15, is also another solution that maximizes the inner optimization. Nevertheless, compared to single-step attacks, as mentioned in Chapter 4, the cost of optimization is proportional to the number of iterations in PGD that can be overextending. Additionally, the optimization is very sensitive to the scale of the gradient. Therefore, in practice, steepest descent is utilized instead of regular gradient descent, and generally, the steepest descent norm is matched with the norm that is minimized for the adversarial attack.

In robustness certification and for the classification models, one needs to perform the integer programming solution using a targeted attack for every possible alternative class label to examine precisely whether any adversarial example exists for a given model. Through this investigation, if none of the optimization objectives is negative for any target class, the classifier is certified to be robust on the particular attack. Nonetheless, certified robustness is not feasible for regression models such as the point processes in the current study.

For instance, considering the Hawkes process, all non-negative values for inter-event times represent a valid adversarial instance, and any time t in the future that fits the intensity functions, is a candidate for the foretasted event. Accordingly, the size of such a set is not exhausted, and finding a solution within such a search space is not computationally feasible.

Interval-propagation-based bounds (IPB) [93] is another approach to solve the optimization problem of the inner optimization problem of the adversarial robustness. Although the IPB approach results in a weak bound for feed-forward networks trained for classification purposes, it has a significant computational advantage and only requires two forward passes through the network. Therefore IPB can be applied to significantly larger feed-forward models and trained with extensive hyper-parameter tuning. Regardless, this technique does not apply to regression models unless one can discretize the problem.

5.2 Adversarial training

In adversarial robustness, as aforementioned, a practical defense approach is adversarial training that linearizes the inner maximization problem [46]. In this low cost, feasible approach, the training dataset is augmented with adversarial examples created by FGSM4.2.3. Nevertheless, results of [60] indicate that in image classification models, it cannot ensure the robustness of multiple adversaries and iterative adversarial attacks, e.g., PGD 4 attack with a significant perturbation factor, $\epsilon = 0.3$, and a high number of iterations, 20+ steps.

In addition to ineffectiveness, another limitation of adversarial training with FGSM is label leaking for significant perturbation factors[59]. In this case, the adversary constructs a minimal set of adversarial samples that raises the risk of overfitting the network. Instead of improving the robustness of the model, it will reduce the performance on actual samples and does not improve the robustness against PGD adversaries.

Therefore, recently, augmenting the training dataset with PGD and other iterative attacks has acquired attention to improving the robustness of models [59]. However, it is noteworthy that the cost of iterative attacks increases with the number of steps, which introduces a severe practical limitation.

Ensemble-adversarial training is another alternative technique that addresses the incapability of FGSM adversarial training against non-FGSM adversarial attacks [74]. In ensemble-adversarial training, the adversarial examples transferred from several fixed pre-trained models are augmented with the original training data to improve the robustness on black-box adversarial attacks.

For adversarial training, scaling to large data sets and weak generalization has remained challenging since the adversarial samples region for each data sample is large and contiguous [74]. Regardless, experimental results of [59] show that the inner optimization of 5.1 is tractable. For the first-order attacks, although there are many regional maxima scattered for each actual sample, they tend to have very well-concentrated loss values. Furthermore, their experiments, in addition to our results in 4, confirm that PGD is a "universal" attack, and robustness against PGD will result in robustness against all attacks solely dependent on first-order information [59] .

5.3 Towards universal robust deep point process

Besides studies on adversarial learning, robustness against adversarial attacks [77], [94] via architecture has recently become a noticeable aspect of the design and development of deep neural networks [95]. Specifically, architecture benchmarking and regularization techniques for robustness [60], [96], [97] has got attention. For example, in [97], the Euclidean (Frobenius) norm of the Jacobian of the network is employed as a regularization term and post-processing technique to improve the robustness of the original model, empirically.

In another study, to fulfill general adversarial training for the neural image classifier, [98] proposes an adversarial training method with a domain adaptation technique. According to the authors, in regular adversarial training, since a specific attack develops the adversarial samples, e.g., FGSM, the defense is targeting that precise attack and is not generalized over further potential attacks [59]. Therefore, from the domain adaptation standpoint, the gap between the distribution of clean data($\mathcal{D}$) and adversarial data($\mathcal{A}$) distributions is considerable. Yet, the perturbation is inapprehensible to human eyes. In [98], such a significant gap has been minimized in high-level representation space by introducing two unsupervised and

supervised domain-adaptation adversarial learning methods. Accordingly, the distribution of adversarial images converges on real images' distribution. The unsupervised component addresses the shift in clean and adversarial data distribution in high-level representation space by aligning the covariance matrices and mean vector of clean and adversarial data. Therefore, they have added the following to the objective function:

$$\begin{aligned} \mathcal{L}_{CORAL}(\mathcal{D},\mathcal{A}) &= \frac{1}{k^2}||C_{\varphi(\mathcal{D})} - C_{\varphi(\mathcal{A})}||_{l_1} \\ \mathcal{L}_{MMD}(\mathcal{D},\mathcal{A}) &= \frac{1}{k}\left|\left|\frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}}\varphi(x) - \frac{1}{|\mathcal{A}|}\sum_{x^{adv}\in\mathcal{A}}\varphi(x^{adv})\right|\right|_1 \\ \mathcal{L}_{UDA}(\mathcal{D},\mathcal{A}) &= \mathcal{L}_{CORAL}(\mathcal{D},\mathcal{A}) + \mathcal{L}_{MMD}(\mathcal{D},\mathcal{A}) \end{aligned} \tag{5.2}$$

Where $C_{\varphi(\mathcal{D})}$ and $C_{\varphi(\mathcal{A})}$ are the clean and the adversarial data's covariance matrices in the logit space.

$\mathcal{L}_{MMD}(\mathcal{D},\mathcal{A})$ is the standard distribution distance metric, Maximum Mean Discrepancy (MMD). And, $\mathcal{L}_{UDA}(\mathcal{D},\mathcal{A})$ is the loss function for Unsupervised Domain Adaptation (UDA) [98].

Ultimately, for adversarial training, they have used a variant of FGSM [59] that avoids *label leaking* effect by using y_{pred} instead of y_{true} in generating adversarial samples as follows:

$$X_{\mathrm{i}}^{adv} = X_{\mathrm{i}} + \epsilon \cdot sign(\nabla_x J(X_{\mathrm{i}}, y_{pred})) \tag{5.3}$$

Where y_{pred} is the predicted class of the model.

Another defense mechanism based on adversarial training is Virtual Adversarial Training (VAT) [99] wherein adversarial samples are produced by the virtual adversarial method [100]. In [72], for original training set $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ and set of attacks and classifiers $(\rho_1, C_1), ..., (\rho_m, C_m)$, the adversarial training set is generated by applying $(\rho_{\mathrm{i}}, C_{\mathrm{i}})$ for $\mathrm{i} = 1, 2, \ldots, m$ to the original training set. As a result, the adversarial training set will have the size of $m \cdot n$ and can be used to train a harder classifier. Note that not all adversarial training approaches utilize the entire dataset to generate adversarial instances, and generally, a subset of the dataset is used for adversarial training.

Besides adversarial robustness, several approaches have been proposed to detect the adversarial samples [101]–[103]. In 2017, [102] proposed feature squeezing as a method to detect adversarial examples. Later, in 2019, [103] examine five defense mechanisms for sequence-based RNN models, including sequence squeezing, adversarial Signatures, adversarial training, RNN Ensemble, and Defense SeqGAN[104]. The proposed methods can be attack-specific or attack agnostic for detecting adversarial examples and making classifiers more robust against adversarial attacks.

As a result of GAN's success in self-supervise modeling, attempts to integrate GAN[105] and adversarial learning in anomaly detection are increasing every day[57], [106]. In an ongoing research project from 2019 to 2021, Zhu et al.[107] employ adversarial learning, Long Short-Term Memory (LSTM), and point process models to detect adversarial anomaly for Spatio-temporal streaming event data. In their framework, the current event's nonlinear dependency regarding past events is modeled using the LSTM. As shown in Figure 5.1, the generator is LSTM. The discriminator is a multi-layer, fully connected network. Finally, The output layer is a $softmax$ that returns the sequence's probability of being an anomaly trajectory.

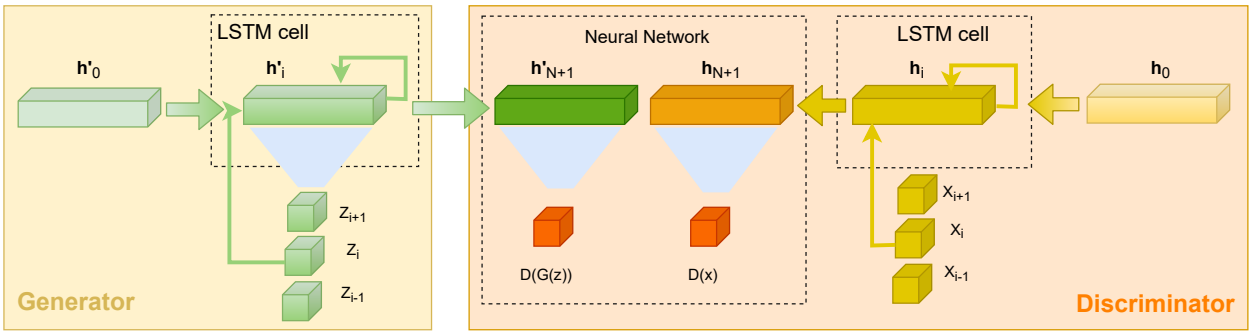


Figure 5.1. Adversarial learning framework proposed by Zhu et al.[107] consists of a LSTM structure and a fully connected neural network.

5.4 Methodology

The robustness of neural point processes to natural shocks and synthetic attacks remains an open problem to date. In the previous Chapter 4, we examined the effect of several existing

and proposed adversarial attacks on predictive performance and parameter modeling of deep point processes. This section presents our methodology to improve the robustness of deep point processes. The discussed adversarial robustness in this section remains within the scope of adversarial attacks proposed in Chapter 4. Specifically, FGSM 4.2.3, PGD 4, R+FGSM 4, momentum iFGSM 4, saliency 4,GD 4, and TM 4 are the precise definition of the attacks our models should be resistant. Since the scope of adversarial attacks is l_∞ attacks, therefore, we focus on the robustness of deep point process models against l_∞ bounded attacks in Chapter 4.

5.4.1 Models

The domain of current chapter is limited to Temporal Hawkes process. In Hawkes process [65]–[67], Hawkes process [65], [67], the event rate is not fixed, but depends on some random inputs, including the history of the process. Hawkes process is a self-exciting process, each arrival increases the rate of future arrivals for some time and is determined by a background Poisson process $\lambda_0(t)$, which is reflecting spontaneous events and at each event in the history a Poisson process g is centered at that event reflecting the increase in the intensity in near future.

$$\lambda(t) = \lambda_0(t) + \sum_{t_\mathrm{i}<t} g(t - t_\mathrm{i}), \tag{5.4}$$

where $\lambda(t)$ donates the event rate at time t [66].

In this work, we are exploring the performance of two deep neural networks implementation of point processes as presented in Chapter 4.

5.4.2 Contribution

The contributions of this section are as follows:

- We evaluate the role of adversarial training in improving the robustness of the deep point processes.
- we propose general point process domain-adopted (GPDA) regularization, which is specifically applicable to point processes. In particular, we differentiate ourselves from

the methods proposed in [98] and [97] by adopting our method to stochastic point processes and using a distinct regularization in the loss function. In addition to the robustness improvement, our method stabilizes the training and improves generalization while the training process remains untouched.

- GPDA regularization is scalable to large networks and uses the available information in the last layer of the model. Therefore, as opposed to other very computationally expensive approaches, there is no need for additional back-propagation in the training step[97] and no need for an additional network[108].

- Ultimately, we leverage the concept of self supervision in generative adversarial networks to propose an adversarial sample detector trained in a Generative Adversarial Network(GAN) setting.

In the next part, we describe each adversarial robustness strategy.

5.4.3 Adversarial training

To the best of our knowledge, adversarial training is not explored for point process models. We explore this phenomenon by augmenting adversarial examples with training data to highlight the robustness of the model against adversarial attacks. Specifically, we augment the clean training set with a set of adversarial samples generated by the PGD 4 against the test set while the perturbation factor, $\epsilon = 0.1$, $\alpha = \epsilon/10$, $k = 10$. Such a procedure maintains the cost of adversarial training minimal and results in feasible adversarial learning.

Due to random initialization and comprehensive coverage of PGD, in addition to adversarial robustness improvement, proposed adversarial training boosts the model's generalization.

5.4.4 General point process domain-adopted (GPDA) regularization

As mentioned in Chapter 4, time-rescaling theorem 4.2.4 and thinned residuals are two methods to evaluate the goodness of fit for some point data to a point process [81], [82]. In

such an examination, the compensator ($\Lambda(\cdot)$) of the point process is essential as it is the core of the time rescaling theorem and residual analysis.

Compensator (Definition) [109]

For a particular point process, the compensator is formulated as

$$\Lambda(t) = \int_0^t \lambda^*(s)ds \tag{5.5}$$

Compensator is a non-decreasing function and exists event if $\lambda^*(\cdot)$, the conditional intensity, does not exists.

Time-rescaling (Theorem) [83]–[85]

Consider $t_1, t_2, , t_k$ as realizations of a particular point process wherein its conditional intensity function is $\lambda^*(\cdot)$ over time $[0, T]$. If $\lambda^*(\cdot) > 0$ over $[0, T]$ and $\Lambda(T) < \infty$, then $\{\Lambda(t_1), \Lambda(t_2), \ldots, \Lambda(t_k)\}$ as transformed points, form a unit rate Poisson process.

The time-rescaling is well-known in point processes and assessing the goodness of fit. In its univariate form, by using this theory, one can transform "any" point process with an integrable conditional intensity function into a unit rate Poisson process [83].

Residual analysis (Theorem) [84], [110], [111]

Consider $t_1, t_2, \ldots$ as event times, and a monotonic, continuous compensator $\Lambda(\cdot)$ such that $\lim_{t\to\infty} \Lambda(t) = \infty$. The transformed points, $\{\Lambda(t_1), \Lambda(t_2), \ldots, \Lambda(t_k)\}$ are the realization of a unit rate Poisson process "if and only if" the original sequence, $\{t_1, t_2, \ldots\}$ is a realization of the point process defined by $\Lambda(\cdot)$.

In the previous chapter, we discussed how the residual analysis is utilized to examine the "goodness of fit". In this assessment, for a fitted model, the inter-event intervals of the orderly point process are resalable, such that the result $(\Lambda(t_{i+1}) - \Lambda(t_i))$ are i.i.d and exponentially distributed with mean 1.

Here, we leverage Theorem 5.4.4 as a regularization term to propose a general point process domain-adopted (GPDA) regularization. In the unguarded model, the loss function is defined as the mean negative log-likelihood (MNLL) such that the log-likelihood function is as follows [68]:

$$LL_{\mathrm{i}} = \sum_{\mathrm{i}} \left[log\left\{ \frac{\partial \Phi}{\partial \tau}(\tau = t_{\mathrm{i}+1} - t_{\mathrm{i}} | h_{\mathrm{i}}) \right\} - \Phi(\tau = t_{\mathrm{i}+1} - t_{\mathrm{i}} | h_{\mathrm{i}}) \right] \tag{5.6}$$

Accordingly, for the guarded model, in addition to maximizing 5.6, we ensure $(\Lambda(t_{\mathrm{i}+1}) - \Lambda(t_{\mathrm{i}}))$ are i.i.d and exponentially distributed with mean 1. Therefore the loss function is defined for a deep temporal point process utilizing GPDA is defined as follows:

$$\mathcal{L} = -\frac{1}{N} \sum_{\mathrm{i}} LL_{\mathrm{i}} + \xi(1 - \frac{1}{N} \sum_{\mathrm{i}} \Phi(\tau = t_{\mathrm{i}+1} - t_{\mathrm{i}} | h_{\mathrm{i}})) \tag{5.7}$$

Where the ξ controls the regularization term. From another standpoint, GPDA regularization forces the deep learning model to fit the data such that the estimated $\Lambda(T)$ is as close as possible to $N(t)$, $\Lambda(T) - N(t) \approx 0$, and for a perfectly fitted model, $N(t) = \Lambda(T)$.

In section 5.6, we illustrate $\xi = 0.1$ is effective in decreasing the effect of adversarial attacks empirically.

5.4.5 Generative adversarial networks

Generative adversarial networks (GANs)[1] are a class of unsupervised deep learning models [105], [112]. Each GAN has two primary components – a Generator Neural Network (G) and Discriminator Neural Network (D). The generator $G(z)$ takes an input, z, a sample from probability distribution $p(z)$ (typically Gaussian), and generates synthetic (fake) data. The discriminator takes as input either a real observation, in our case inter-event time sequence, or synthetic output from the generator, and determines through binary classification whether the input is real or fake. The two networks are trained jointly, where alternating gradient descent steps are made to I) train the generator by freezing the weights of the discriminator

[1] © 2020 IEEE. Reprinted, with permission, from S. Khorshidi, G. Mohler and J. G. Carter, Assessing GAN-based approaches for generative modeling of crime text reports, IEEE International Conference on Intelligence and Security Informatics (ISI), Nov, 2020

and then update the weights of the generator so as to increase the probability that fake data is labeled real and II) train the discriminator to correctly classify real and fake data. Upon successful training, the two networks reach an equilibrium in what can be viewed as a minimax game [113].

Much of the success of GANs has occurred for continuous data in the context of images [114][115] [116][104] [117], and using GAN on discrete data such as sequence-based data is a challenging task.

Previously, in [118], we examined the effectiveness of GANs in generating domain-specific text. In particular, we utilized several GAN frameworks to generate and represent crime report text data. Here, we employ GAN for an different purpose, to discriminate adversarial sequences. In particular, we propose a novel GAN-based point process adversarial sample detection (PointGAN) in this section, as presented in Figure 5.2. Specifically, we propose a GAN framework utilizing LSTM for both the Generator and Discriminator networks. In this framework, we train our GAN only on the real dataset, no adversarial, in the training phase. Subsequently, the trained discriminator acts as an adversarial sample detector that distinguishes the adversarial samples from the real samples in the detection phase.

Same as the deep point process models, here, GAN is provided with sequences of inter-event times, $t_{\mathrm{i}} - t_{\mathrm{i}-1}$, and the LSTM architecture with 100 units [119] captures the temporal relation in the input sequence and takes into account the latent interactions between them. To improve the performance of the generator, the latent space has size of 5. In the training phase, similar to the original GAN, the discriminator and generator networks are playing a *minmax* game while generator tries to fool discriminator by generating more realistic data samples by getting feedback from the discriminator and finally they both reach an equilibrium[113].

Goodfellow et al. [105] have proposed the minimax formulation for an optimal discriminator to minimize the Jensen Shannon Divergence (JSD) between the generator's output distribution and the actual data distribution as follows:

$$\min_G \max_D V(D,G) = \underset{x\sim P(x)}{E} [\log D(x)] + \underset{z\sim P(z)}{E} \log(1 - D(G(z))) \tag{5.8}$$

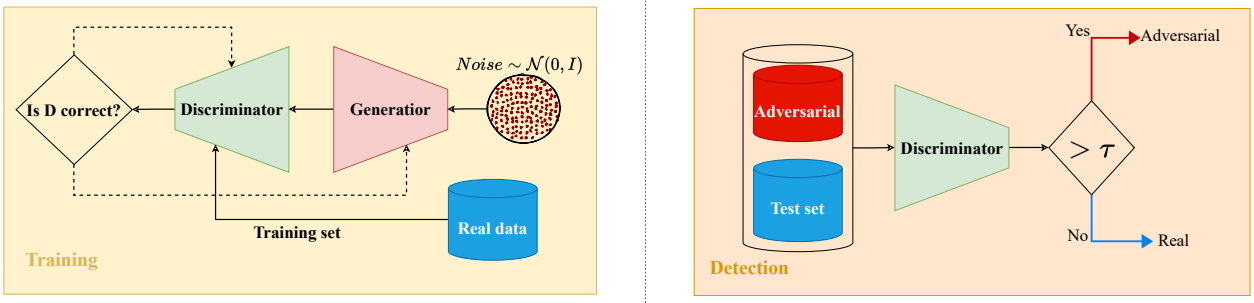


Figure 5.2. GAN-bases adversarial sample detection (PointGAN) framework. In training stage (Left), the discriminator is trained on training-set of real data (No adversarial) and the generated samples from the Generator. In Detection phase (Right), the performance of pre-trained discriminator is evaluated on a mixed data, containing both real and adversarial sequences. τ is the adversarial probability threshold.

In the current work, we follow the same optimization approach. For the discriminator, similar to [106], we define the loss function as

$$D_{loss} = \frac{1}{m}\sum_{\mathrm{i}=1}^{m}[logD(x_{\mathrm{i}}) + log(1 - D(G(z_{\mathrm{i}})))] \Leftrightarrow \frac{1}{m}\sum_{\mathrm{i}=1}^{m}[-logD(x_{\mathrm{i}}) - log(1 - D(G(z_{\mathrm{i}})))] \tag{5.9}$$

And for the Generator,

$$G_{loss} = \sum_{\mathrm{i}=1}^{m}[log(1 - D(G(z_{\mathrm{i}})))] \Leftrightarrow \sum_{\mathrm{i}=1}^{m} log(-D(G(z_{\mathrm{i}}))) \tag{5.10}$$

Where x_{i}, $\mathrm{i} = 1, \ldots, m$ are the the real samples and $G(z_{\mathrm{i}})$ for $\mathrm{i} = 1, \ldots, m$ are the synthetic samples generated by the generator. In this framework, the discriminator averages the negative cross-entropy between its predictions and sequence labels. Simultaneously, the generator aims to mislead the discriminator by generating realistic samples. In section 5.6, we present some generated samples by the generator at early and late stages that confirm the generator can generate realistic samples.

5.4.6 Baseline and Metrics

The baseline corresponds to the standard case when a model is built without adversarial assumptions and has no guard against potential adversarial attacks. We compare the performance of each defense mechanism in improving the robustness against adversarial attacks discussed in Chapter 4. Specifically, to quantify the performance of each mechanism, we compare the performance of our models utilizing robustness mechanisms in the presence of adversarial attacks using the following metrics:

- **Mean Negative Log Likelihood** Our first metric in the mean negative log likelihood which is also part of the loss function of neural network models in this work. Since NLL depends on the predictive uncertainty, it's a reasonable metric in evaluating predictive uncertainty [79], [80].

- **Mean Absolute Error**

$$MAE = \frac{1}{K}\sum_{k=1}^{K}||T(x_k) - y_k||_1 \tag{5.11}$$

$$MAE_{adv} = \frac{1}{K}\sum_{k=1}^{K}||T(x_k^{adv}) - y_k||_1 \tag{5.12}$$

- **Fooling Error**[71]

$$FE = \frac{1}{K}\sum_{k=1}^{K}||T(x_k^{adv}) - T(x_k)||_q \tag{5.13}$$

- **Symmetric Mean Accuracy Percentage Error** [71]

$$SMAPE = \frac{2}{K_+}\sum_{k=1}^{K_+}\frac{||T(x_k^{adv}) - y_k||_q - ||T(x_k) - y_k||_q}{||T(x_k^{adv}) - y_k||_q + ||T(x_k) - y_k||_q} \tag{5.14}$$

  Where q norm in FE and SMAPE metrics, must match the l_p norm employed in generating adversarial attacks and SMAPE is limited to the K_+ positive elements in the summation.

- **Local Loss sensitivity** Theoretically, a critical characteristic of deep learning models that makes them susceptible and puts them at higher risk of adversarial attacks is the degree of smoothness in the loss surface, as a raggeder loss surface is at a higher risk. The local loss sensitivity (LLS) quantifies loss surface smoothness by assessing its Lipschitz continuity constant, which is a strong form of uniform continuity for functions and measures the largest variation of a function for a slight modification in the input [120]. In deep learning modeling, LLS can be estimated based on the gradient of the models [72], [121]. LLS is defined as:

$$LLS(T, X, y) = \frac{1}{n}\sum_{\mathrm{i}=1}^{n}||\nabla_x \mathcal{L}(x_{\mathrm{i}}, y_{\mathrm{i}})||_2 \tag{5.15}$$

  Where here, T is the deep learning model, $X = (x_1, \ldots, x_n)$ is the test set, and y_{i} is the correspondent target value. In this metric, lower LLS means more robust the model is

against adversarial attacks. Additionally, as it can be inferred from (5.15), LLS is an attack-independent metric and provides insight into the model's characteristics.

It's notable that the q norm in FE and SMAPE metrics, must match the l_p norm employed in generating adversarial attacks.

5.5 Data

Similar to [43], [68], experiments are run on the following point process datasets. We split each dataset into train and test sets. The train set is then used to estimate the model's parameters in the training phase, and the model's performance on the test set is quantified by the evaluation metrics.

5.5.1 Hawkes Process

Similar to Chapter 4, to asses the effectiveness of defense mechanisms, we have simulated [66] 100,000 event times from the Hawkes process with the sum of exponentials kernel, and the conditional intensity function is given by:

$$\lambda(t|H_t) = \mu + \sum_{\mathrm{j}}^{M} \sum_{t_\mathrm{i}<t} \alpha_\mathrm{j}\beta_\mathrm{j} \exp\{\beta_\mathrm{j}(t - t_\mathrm{i})\} \tag{5.16}$$

Where $M = 1$, $\mu = 0.2$, $\alpha_1 = 0.8$, and $\beta_1 = 1.0$.

In equation 5.16, μ is the background intensity, α is adjacency and represents the jump in intensity after the arrival of a new event, and β is the intensity decay rate after arrival.

In the exponentially decaying Hawkes process, $\frac{\alpha}{\beta}$ is known as *branching ratio* where the value of $\frac{\alpha}{\beta}$ is the declaration of whether or not the Hawkes process explodes. The Hawkes process can be in three regimes depending on the branching ratio.

Specifically, if the branching ratio is 0, the model will become a non-homogeneous Poisson process since the initial immigrants will trigger no further events. Indeed, this is Hawkes process as a generalization of the Poisson process that considers the time and history of a process [87]. For $\frac{\alpha}{\beta}$ ratio, the critical threshold is 1, which splits the process into subcritical

($\frac{\alpha}{\beta} < 1$) and supercritical ($\frac{\alpha}{\beta} > 1$) regimes. If $\frac{\alpha}{\beta} > 1$ for a sustained amount of time, the modeled intensity may explode.

Finally, if the background intensity is a constant and the process is in the subcritical regime ($\frac{\alpha}{\beta} < 1$), the branching ratio can be seen as a probability [88].

5.5.2 Crimes and Covid-19

To assess the performance of models concerning non-stationary changes, such as the effect of the Covid-19 pandemic on reported crimes in Chicago, we use the reported crimes to the Chicago police department from 2016 to 2018 as training, and crimes in 2019 and 2020 as test set, separately[2].

5.6 Results and discussion

This section presents the effectiveness of proposed defense mechanisms in improving the robustness of both fully neural network (NN) and exponential hazard (EXP) models against adversarial attacks. As aforementioned, we proposed two approaches to improve models' robustness: adversarial training and general point process domain-adopted (GPDA) regularization. In all presented results related to adversarial training, %10 of the original training set is augmented with adversarial sets crafted via PGD algorithm as we discussed in Section 5.4.3. Moreover, for results on GPDA, we have considered $\xi = 0.1$, which we found to be effective. The adversarial attacks we aimed to defense against are previously proposed attacks in Chapter 4. For the exponential hazard (EXP) model, since adversarial training destabilizes the training process, GPDA is the only experimented defense mechanism.

Additionally, this section presents the effectiveness of defense mechanisms in reducing the effect of non-stationary abrupt changes on the models' performance using the COVID-19.

Regarding macro robustness, we illustrate the capability of both defense mechanisms in improving the generalization as well as adversarial robustness. Specifically, we show how proposed defense mechanisms are able to reduce and neutralize the effect of adversarial attacks on the modeled Hawkes process's parameters.

[2]↑https://data.cityofchicago.org/Public-Safety/Crimes-Map

Finally, we present the performance of poinGAN as a self-supervised adversarial sample detection compared to well-known supervised methods.

In Table 5.1, we present the effect of each defense mechanism on the model's generalization where the performance of models on real (clean) data has been provided. According to the result, both defense mechanisms reduce the prediction uncertainty on real data, as shown in the MNLL metric. Additionally, Adversarial training (AT) and GPDA effectively reduce models' local loss sensitivity (LLS). However, since GPDA illustrates more significance than AT, we conclude the effect of the merged defense mechanism on LLS is because of PGDA. Finally, although the MAE of guarded models on real data is enhanced by each defense mechanism individually when both defense methods are utilized together, the NN model's MNLL and MAE are not improved compared to the unguarded model.

Table 5.1. Adversarial training (AT) and GPDA effect on regular performance of the fully neural network model (NN) and Exponential (EXP) model.

Model	**GPDA**	**AT**	**MNLL**$(\mathrm{e}-01)$	**MAE**$(\mathrm{e}-01)$	**LLS**$(\mathrm{e}-05)$
NN	–	–	$4.10 \pm 2.82\mathrm{e}-03$	$7.77 \pm 1.72\mathrm{e}-04$	$7.55 \pm 2.85\mathrm{e}-06$
	–	✓	$4.06 \pm 2.02\mathrm{e}-03$	$7.68 \pm 1.38\mathrm{e}-04$	$6.04 \pm 1.81\mathrm{e}-06$
	✓	–	$4.04 \pm 1.45\mathrm{e}-02$	$7.70 \pm 4.39\mathrm{e}-04$	$0.00 \pm 3.27\mathrm{e}-07$
	✓	✓	$4.30 \pm 2.66\mathrm{e}-02$	$7.80 \pm 4.58\mathrm{e}-04$	$0.00 \pm 0.00\mathrm{e}+00$
EXP	–	–	$4.63 \pm 7.15\mathrm{e}-03$	$8.01 \pm 1.26\mathrm{e}-02$	$2.87 \pm 3.34\mathrm{e}-06$
	✓	–	$4.41 \pm 4.69\mathrm{e}-03$	$7.89 \pm 1.28\mathrm{e}-03$	$0.13 \pm 1.22\mathrm{e}-07$

In Table 5.2 and Table 5.3, we present the effectiveness of defense mechanisms against adversarial attacks. We evaluate their capability in these tables when employed independently and merged. In Figure 5.3 and Figure 5.4 the predictive performance of the models when they have no guard against the adversarial attack has been represented as compared to employing defense mechanisms. According to experimental results, proposed defense mechanisms are reducing the error in prediction. Effectiveness of adversarial training in reducing the prediction uncertainty of the fully Neural network (NN) model in presence of first order attacks, confirms PGD to be universal first-order white-box attack. Albeit GPDA doesn't improve the MNLL of models in presence of all adversarial attacks, it's competent in reducing the effect of adversarial attacks on fooling error (FE) and SMAPE.

Table 5.2. Effectiveness of adversarial training (AT) and GPDA in improving the robustness of the fully neural network model (NN) against each attack.

Attack	**GPDA**	**AT**	**MNLL**$(\mathrm{e}-01)$	**MAE**$(\mathrm{e}-01)$	**FE**$(\mathrm{e}-02)$	**SMAPE**$(\mathrm{e}-01)$
PGD	–	–	5.90	8.09	3.64	3.13
	–	✓	5.83	8.03	3.06	2.71
	✓	–	6.46	8.02	2.95	2.93
	✓	✓	6.97	8.01	2.27	2.53
iFGSM	–	–	6.54	8.21	4.75	3.66
	–	✓	6.12	8.11	3.67	2.92
	✓	–	7.76	8.07	2.98	3.20
	✓	✓	7.00	8.34	4.93	3.02
R+FGSM	–	–	6.04	8.11	3.83	3.26
	–	✓	5.88	8.04	3.19	2.70
	✓	–	6.62	8.04	2.98	2.89
	✓	✓	6.54	8.02	2.80	2.61
miFGSM	–	–	6.64	8.20	4.75	3.67
	–	✓	6.30	8.10	3.88	3.05
	✓	–	7.32	8.10	3.58	3.21
	✓	✓	6.94	8.18	4.11	2.95
FGSM	–	–	6.56	8.18	4.75	3.60
	–	✓	6.27	8.11	3.88	3.10
	✓	–	7.92	8.09	2.99	3.18
	✓	✓	7.06	8.08	3.29	2.81
SM	–	–	4.64	7.91	1.58	1.56
	–	✓	4.89	7.86	1.04	1.24
	✓	–	5.16	7.88	1.14	1.40
	✓	✓	5.86	7.99	1.21	0.986
GD	–	–	6.73	7.85	2.43	3.33
	–	✓	5.20	7.87	2.99	3.30
	✓	–	6.79	7.85	2.06	3.39
	✓	✓	6.14	7.90	2.22	3.24
TM	–	–	5.98	8.36	21.2	7.08
	–	✓	6.52	8.38	21.2	6.89
	✓	–	7.08	8.23	14.1	6.30
	✓	✓	7.42	8.24	15.7	6.12

Table 5.3. Effectiveness of GPDA in improving the robustness of the exponential model (EXP) against each attack.

Attack	**GPDA**	**MNLL**(e − 01)	**MAE**(e − 01)	**FE**(e − 02)	**SMAPE**(e − 01)
PGD	–	5.38	8.30	2.62	1.76
	✓	5.54	8.04	1.00	1.24
iFGSM	–	5.67	8.26	3.66	2.32
	✓	5.41	8.06	1.25	1.01
R+FGSM	–	5.43	8.19	2.89	1.93
	✓	5.24	8.01	1.41	1.29
miFGSM	–	5.65	8.30	3.57	2.16
	✓	5.59	7.96	1.16	1.32
FGSM	–	5.72	8.33	4.01	2.28
	✓	5.28	8.11	1.46	1.25
SM	–	4.76	8.06	0.673	0.805
	✓	5.12	7.99	0.287	0.707
GD	–	5.28	8.16	4.73	2.24
	✓	5.20	8.02	3.01	2.16
TM	–	6.67	8.72	28.6	6.27
	✓	7.21	8.29	16.4	5.41

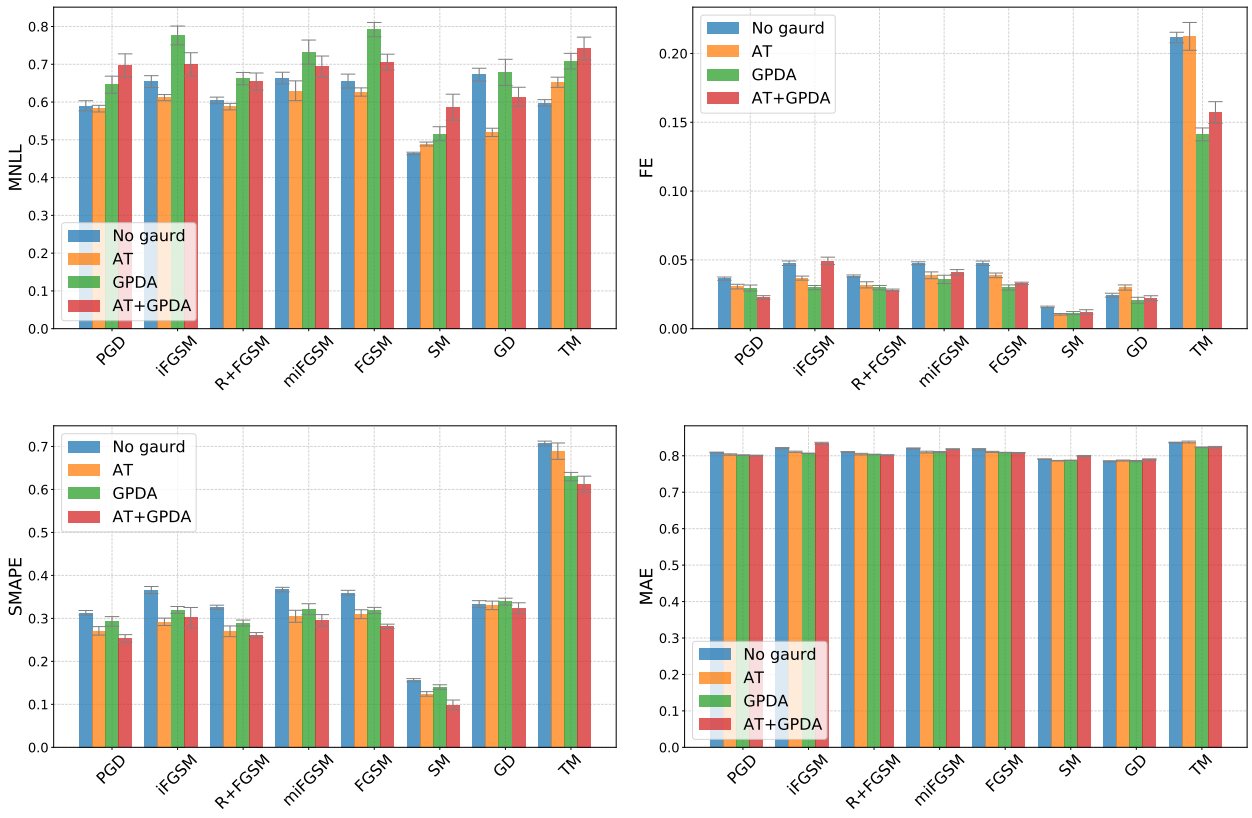


Figure 5.3. Effectiveness of each guard on the performance of each attack against the fully neural network (NN) model, in terms of MNLL (top left), FE (top right), SMAPE (bottom left), and MAE (bottom right).

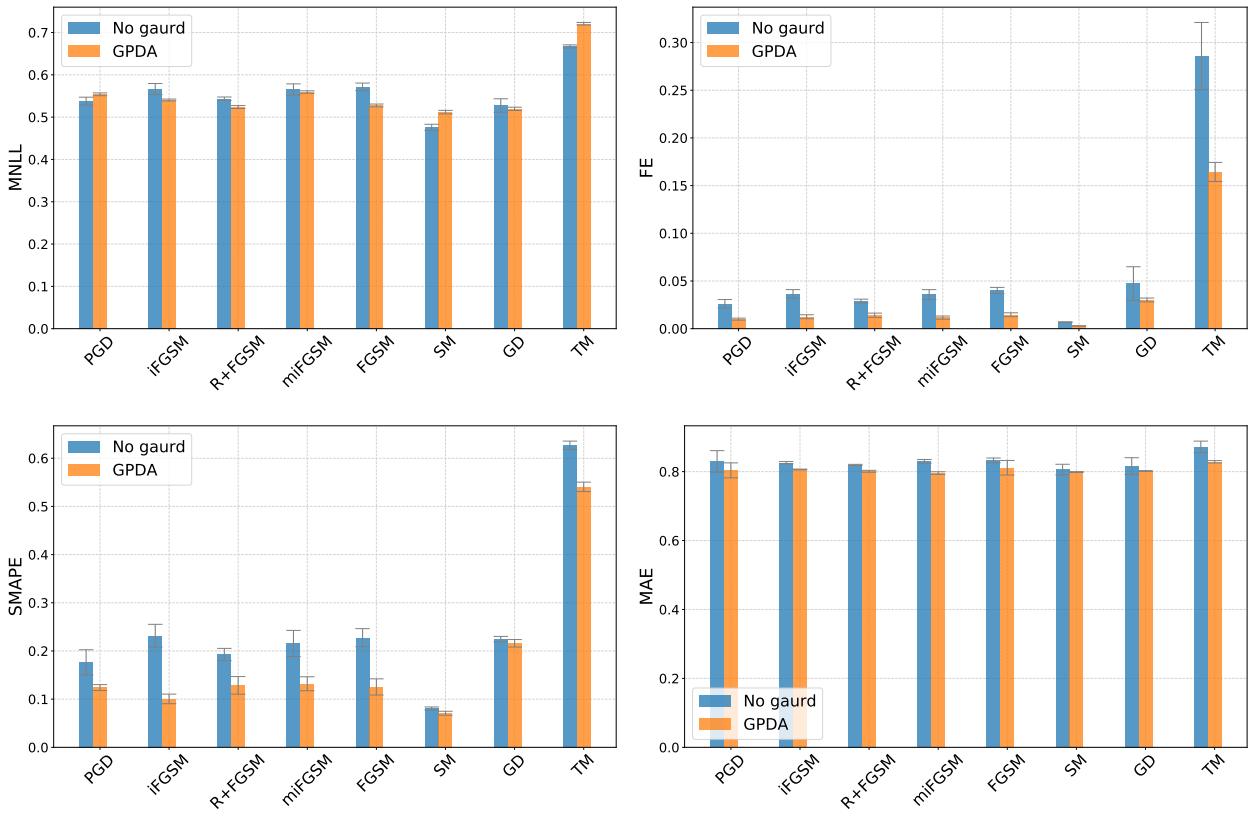


Figure 5.4. Effectiveness of each guard on the performance of each attack against the exponential (EXP) model, in terms of MNLL (top left), FE (top right), SMAPE (bottom left), and MAE (bottom right).

Considering the danger of adversarial attacks to parametric modeling of point processes, as discussed in Chapter 4, we assess the capability of the defense mechanisms in eliminating such a risk. Specifically, in Table B.3 and Table B.4 presented in Appendix B, we present the beneficial effect of defense mechanisms utilized individually and together on the Hawkes process's parameters in the presence of adversarial attacks.

Accordingly, in Figure 5.5, Figure 5.6, and Figure 5.7 we illustrate the effectiveness of GPDA, AT, and GPDA+AT in reducing the effect of white-box adversarial attacks, on predicted Hawkes process parameters by the fully neural network model (NN). Likewise, in Figure 5.8, we demonstrate the effectiveness of GPDA in eliminating the effect of adversarial attacks on predicted Hawkes process parameters by the Exponential hazard model (EXP). Note, in these figures we have not presented all first-order adversarial attacks since the are tighten to PGD.

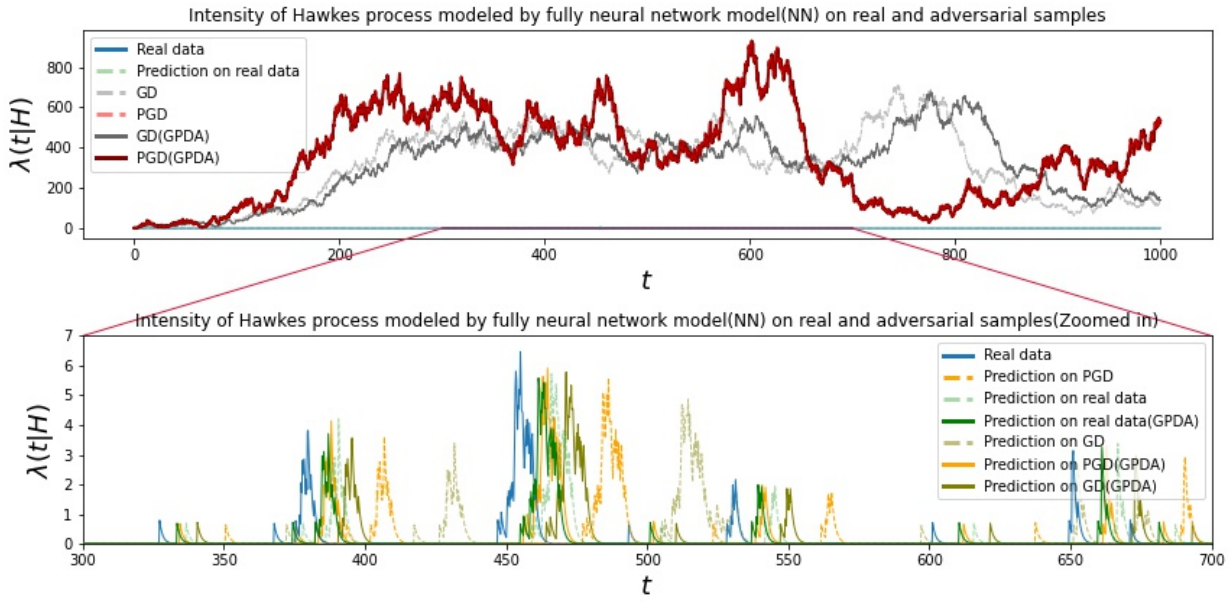


Figure 5.5. GPDA effectiveness in diminishing adversarial attack's macro effect on the fully neural model (NN) parametric modeling performance. Top: Conditional intensity of real events (Blue), predicted events (Green), adversarial events generated by PGD attack (Red), and GD(Gray), respectively. Bottom: Enlarged view. Note that adversarial samples' intensity, $\lambda(t|H)$, range is far from the real and predicted samples. Here, we represent $\lambda(t|H)$ of predicted events in presence of GD attack in "Olive". Intensity functions in the unguarded phase are presented in $--$ lines.

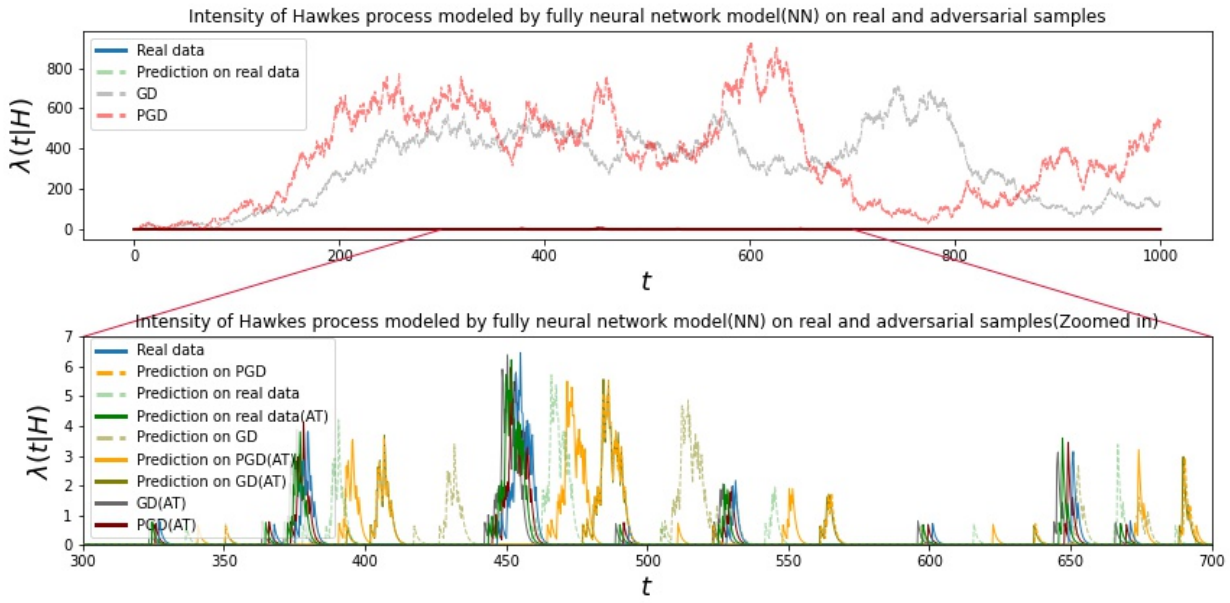


Figure 5.6. Adversarial training (AT) effectiveness in diminishing adversarial attack's macro effect on the fully neural model (NN) parametric modeling performance. Top: Conditional intensity of real events (Blue), predicted events (Green), adversarial events generated by PGD attack (Red), and GD(Gray), respectively. Bottom: Enlarged view. Note that in the unguarded situation, the adversarial samples' intensity, $\lambda(t|H)$, range is far from the real and predicted samples. Here, we represent $\lambda(t|H)$ of predicted events in the presence of GD attack in "Olive". Intensity functions in the unguarded phase are presented in $--$ lines.

In these figures, we present the conditional intensity of samples in the absence of any defense mechanism in dashed lines ($--$). From Figure 5.5, GPDA can reduce adversarial effects on predicted parameters and improve the model's generalization on real samples as "Prediction on real data" is approaching real data intensity. Note that GPDA does not affect first-order adversarial samples' intensity and Hawkes parameters.

As shown in Figure 5.6, utilizing adversarial training (AT) and the effect of predicted Hawkes processes tighten the intensity of adversarial samples to real samples. Also, AT can improve the model's generalization on real samples. However, it's less capable of reducing the effect of first-order adversarial attacks on predicted samples, as "Prediction on PGD" in GPDA is closer to real samples' intensity.

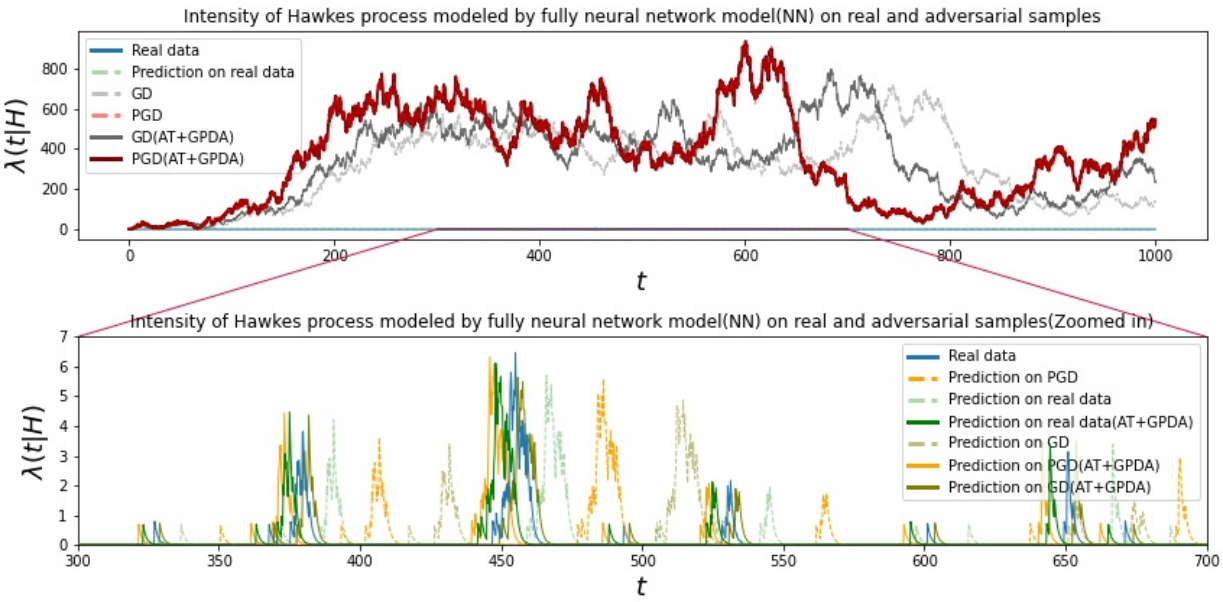


Figure 5.7. GPDA and AT joint effectiveness in reducing adversarial attack's macro effect on the fully neural model (NN) parametric modeling performance. Top: Conditional intensity of real events (Blue), predicted events (Green), adversarial events generated by PGD attack (Red), and GD(Gray), respectively. Bottom: Enlarged view. Note that in the unguarded situation, the adversarial samples' intensity, $\lambda(t|H)$, range is far from the real and predicted samples. Here, we represent $\lambda(t|H)$ of predicted events in the presence of GD attack in "Olive". Intensity functions in the unguarded phase are presented in $--$ lines.

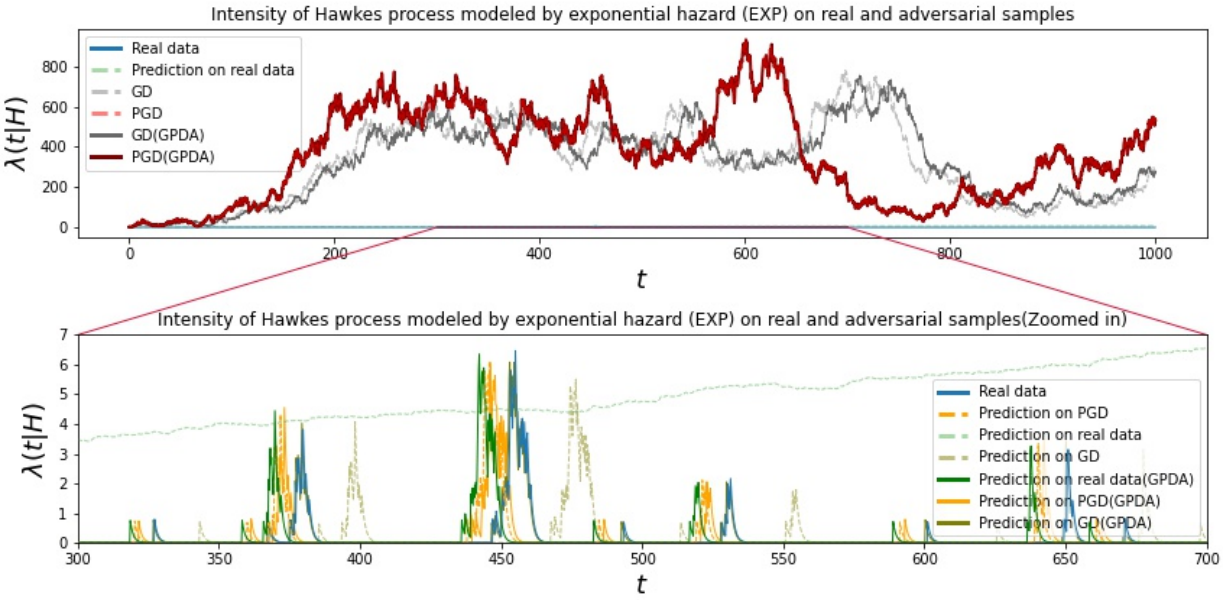


Figure 5.8. GPDA effectiveness in reducing the adversarial attack's Macro Effect against the exponential hazard (EXP) model's parametric modeling performance. Top: Conditional intensity of real events (Blue), predicted events (Green), adversarial events generated by PGD attack (Red), and GD(Gray), respectively. Bottom: Enlarged view. Note that adversarial samples' intensity, $\lambda(t|H)$, range is far from the real and predicted samples. Here, we represent $\lambda(t|H)$ of predicted events in presence of GD attack in "Olive". Intensity functions in the unguarded phase are presented in $--$ lines.

Finally, by employing both adversarial training and GPDA in defending the fully neural network (NN) model shown in Figure 5.7, we can maximize the usefulness of defense approaches in eliminating the threat of the adversarial attacks to parametric modeling as well as enhancing the model's generalization.

Similarly, we present the positive effect of utilizing GPDA as a defense mechanism in both improving the model's generalization an reducing the adversarial effect on parametric modeling of Exponential hazard model (EXP).

To evaluate the effectiveness of the defense mechanisms in reducing the transferability of white-box attacks against NN and EXP models, for each attack, we perform attacks against independently trained networks with more complexity and larger architecture and then deploy the adversarial set to our original targeted models and compare the performance with unguarded models. Similar to the transferability analysis in the previous chapter, to increase the complexity, we try hyper-parameters as discussed in the transferability section 4. The "RNN units" is the only available hyper-parameter for the exponential hazard model. In Figure 5.9 and Figure 5.10, we illustrate the transferability power of each attack against our models in the presence of defense mechanisms. Considering readability of plots, not all white-box attacks are presented. The presented results imply that the proposed defense mechanisms can relieve the risk of transferability of white-box attacks to be utilized as black-box attacks.

In the following, in Table 5.4, we illustrate the defense mechanism's ability to improve the models' performance in the presence of non-stationary changes in data. According to the results, GPDA drives both the exponential and NN models to become more robust. We should note that the COVID-19 pandemic that causes non-stationary changes started in March 2020. Thus, although the defense mechanisms have not improved the NN's performance on 2019's reported crimes, PGDA still improved its performance concerning non-stationary abrupt changes in 2020. In all other cases, including EXP in 2019, GPDA has boosted the LLS, which results in adversarial robustness, improves the predictive performance as shown by MAE, and reduces predictive uncertainty as reported by MNLL; in other words, enhancing generalization.

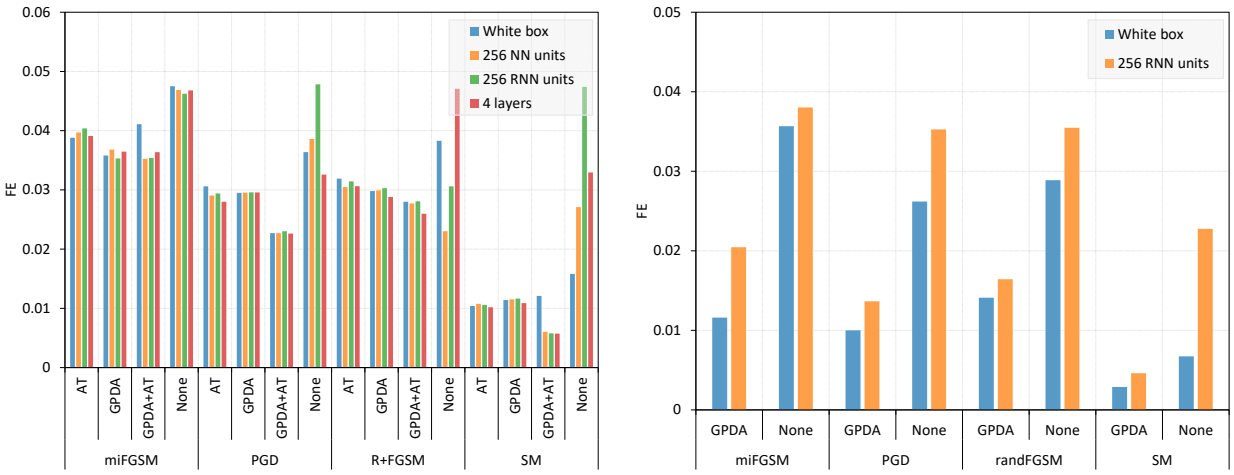


Figure 5.9. Effectiveness of defense mechanisms in reducing fooling error (FE) of each attack in the white box and semi-black-box settings against (left) fully neural model (NN) and (right) exponential kernel hazard model (EXP). The attacker has either 4 layers, 256 neural units, or 256 RNN units for the semi-black-box setting.

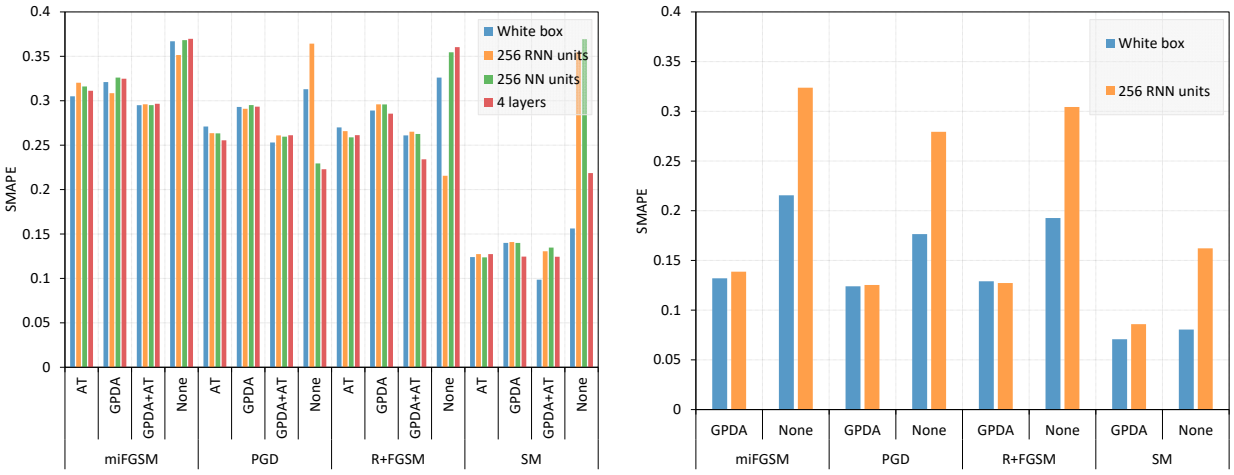


Figure 5.10. Effectiveness of defense mechanisms in reducing symmetric Mean Accuracy Percentage Error (SMAPE) of each attack in the white box and semi-black-box settings against (left) fully neural model (NN) and (right) exponential kernel hazard model (EXP). The attacker has either 4 layers, 256 neural units, or 256 RNN units for the semi-black-box setting.

Table 5.4. Effectiveness of adversarial training and GPDA in improving the robustness of the models on non-stationary changes in reported crimes in 2019 and 2020.

Data	**Model**	**GPDA**	**AT**	**MNLL**$(\mathrm{e}+00)$	**LLS**$(\mathrm{e}-08)$	**MAE**$(\mathrm{e}+02)$
Crimes (2019)	NN	–	–	**5.08**	8.66	**0.96**
		–	✓	5.16	6.04	0.99
		✓	–	5.23	1.23	1.01
		✓	✓	5.27	**0.836**	1.05
	EXP	–	–	5.67	1.02	1.11
		✓	–	**5.64**	**0.234**	1.11
Crimes (2020)	NN	–	–	5.44	6.38	1.31
		–	✓	5.41	6.48	1.30
		✓	–	**5.22**	**3.23**	**1.25**
		✓	✓	5.49	1.68	1.30
	EXP	–	–	5.929	1.60	1.46
		✓	–	**5.85**	**0.355**	**1.44**

$*$ Non-stationary changes in data are not measurable with FE and SMAPE metrics.

As we discussed before, the limitation of the Exponential hazard model (EXP) model is its explicit functional form for the time course of the hazard function [68]. Such an assumption diminishes the expressiveness of the exponential hazard (EXP) model in modeling the general dependency of the event occurrence on the past points, as confirmed by the results on adversarial attacks and non-stationary abrupt changes, Covid-19 effect on crimes. A straightforward approach to reclaim such a dependency is employing a more complex functional form of hazard function to generalize the model, which to the same extent increases the model complexity and is intractable because of the log-likelihood function, specifically in deep learning modeling as suggested by [68]. Due to the unrestricted context of the proposed GPDA in the current work and according to the presented results on predictive performance and parametric modeling, the GPDA approach is able to improve the generalization and robustness of the Exponential hazard (EXP) model without any additional requirement or other complex function. Ultimately, we examine the ability of the generative adversarial network (GAN) to generate point process sequences and provide a self-supervised trained adversarial detector. As mentioned in the previous section, we have trained the adversarial detector in GAN-setting. To ensure the quality of generated sequences by the generator, we present two sets of generated samples at early and late stages by the generator of pointGAN in Figure 5.2. As illustrated, at later stages, the generated samples look realistic, which implies the generator of pointGAN has learned the latent space distribution of the real clean dataset.

Finally, in Table 5.5 and Table 5.6 provided in Appendix B, we present the capability of pointGAN's discriminator in detecting the adversarial samples from each attack. In the interest of completeness, in Figure 5.11 and Figure 5.12, we illustrate the overall performance of the pointGAN detector in detecting adversarial samples compared to the support vector machine (SVM) and K-nearest neighborhood (KNN) classifiers. In this experiment, the discriminator threshold (τ) of the pointGAN discriminator is 0.6, and both SVM and KNN classifiers have been well-tuned prior to experiment. The figure shows that the pointGAN detector outperforms SVM and KNN in detecting adversarial attacks against NN and EXP models. The consistent performance of pointGAN in detecting attacks suggests that all first-order white-box attacks share similar latent distribution. Regarding adversarial de-

tection, high recall and precision are preferred, and according to the experimental results, the proposed pointGAN detector achieves high consistent recall and precision. At the same time, KNN and SVM show varying performance depending on adversarial attacks. It's noteworthy that adversarial sample detection is considered an imbalanced classification problem. Thus "Accuracy" is not a suitable metric of evaluation. The detailed results are provided in Appendix B.

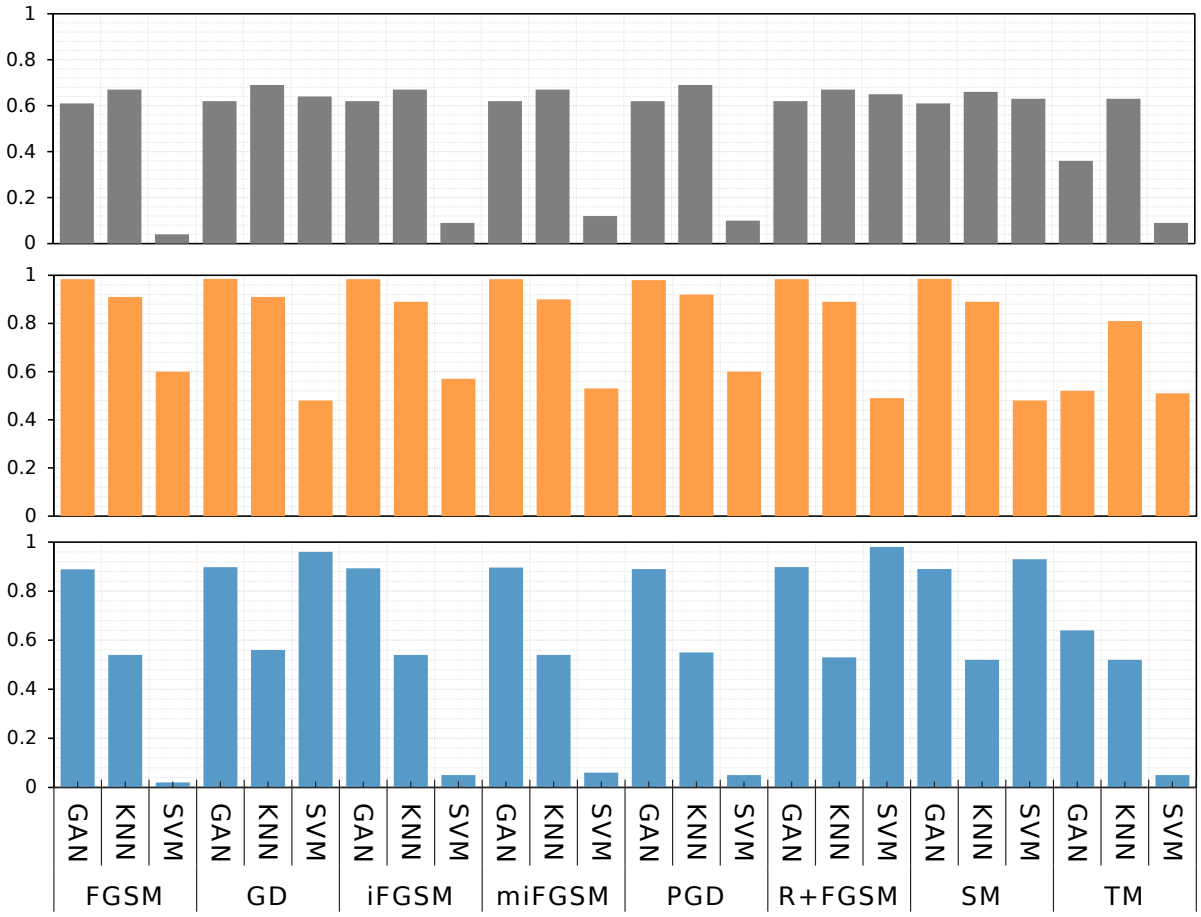


Figure 5.11. Overall performance of PointGAN compared to KNN and SVM in detecting adversarial samples against fully neural network model (NN). Bottom: Recall. Middle: Precision. Top: F1 score.

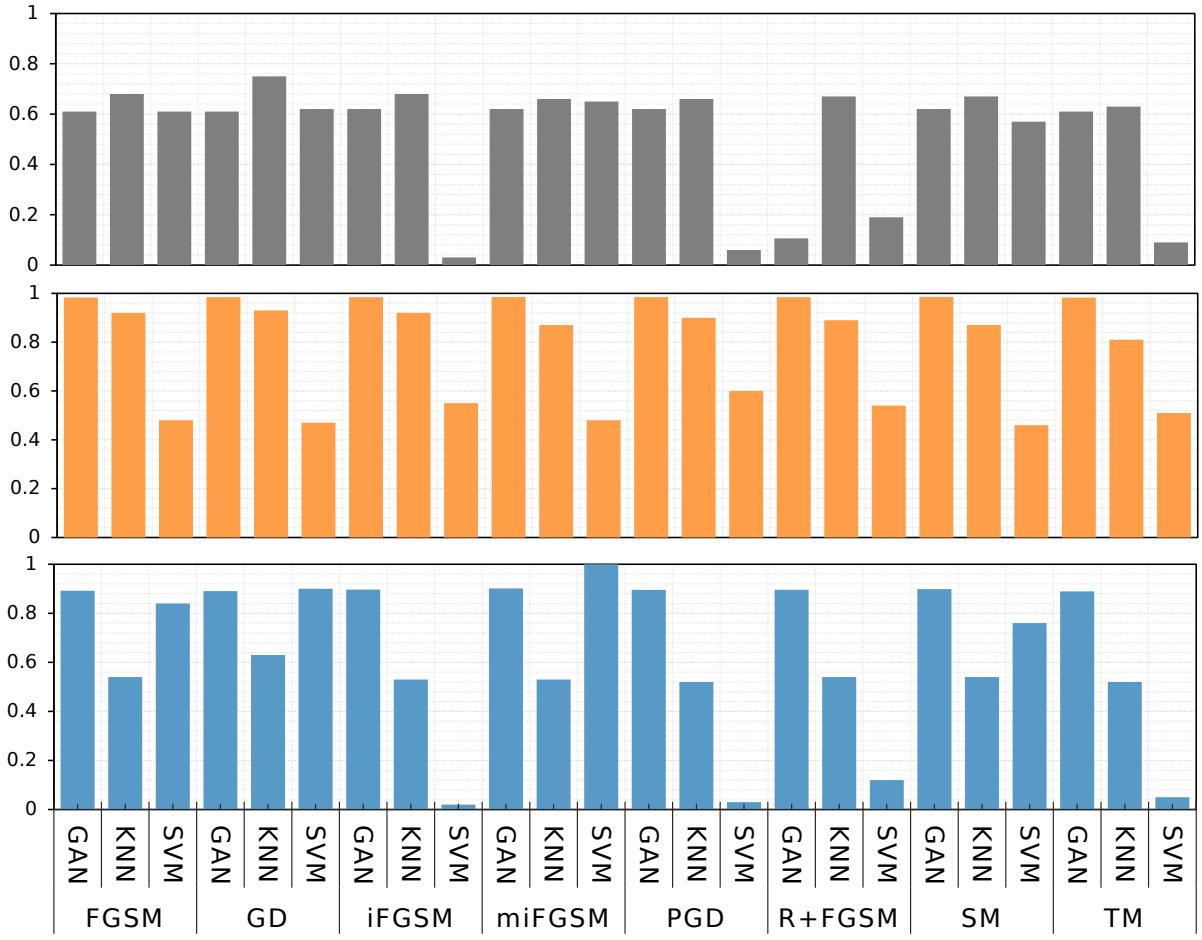


Figure 5.12. Overall performance of PointGAN compared to KNN and SVM in detecting adversarial samples against Exponential hazard model(EXP). Bottom: Recall. Middle: Precision. Top: F1 score.

Table 5.5. PointGAN attack detection performance on different adversarial attacks against the NN model

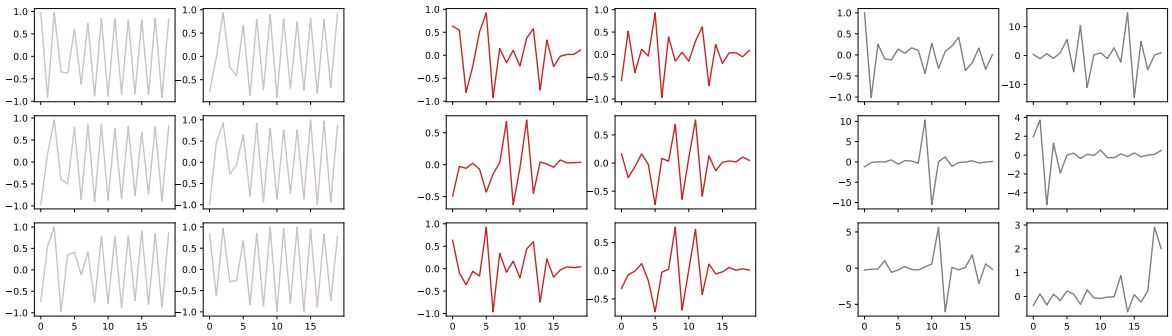
Attack	Accuracy	Precision	Recall	F1 score
PGD	55.99	0.98	0.89	0.62
iFGSM	56.26	0.98	0.89	0.62
R+FGSM	55.95	0.98	0.90	0.62
miFGSM	56.01	0.98	0.90	0.62
FGSM	55.92	0.98	0.89	0.61
SM	56.08	0.99	0.89	0.61
GD	56.30	0.98	0.90	0.62
TM	57.18	0.52	0.64	0.36

∗ Reported metrics are averaged over 1000 epochs

Table 5.6. PointGAN attack detection performance on different adversarial attacks against the EXP model

Attack	Accuracy	Precision	Recall	F1 score
PGD	56.18	0.98	0.90	0.62
iFGSM	55.93	0.98	0.90	0.62
R+FGSM	56.28	0.98	0.90	0.62
miFGSM	55.71	0.99	0.90	0.62
FGSM	55.99	0.98	0.89	0.61
SM	56.05	0.99	0.90	0.62
GD	56.33	0.98	0.89	0.61
TM	55.78	0.98	0.89	0.61

∗ Reported metrics are averaged over 1000 epochs



(a) Generated (iteration 51) (b) Generated (iteration 825) (c) Original samples

Figure 5.13. Two sets of generated instances at two training stages: GAN-generated samples at the early stage are relatively random. In contrast, generated samples at later stages look more realistic and close to the original ones.

6. SUMMARY

Temporal point processes are powerful tools in modeling and inferring the occurrence of events. Point processes and in specific Hawkes process are employed in sensitive and security-related fields, such as criminology. Additionally, due to the advances of neural network and deep learning methods, recently neural network modeling of point processes have received the attention. However, up to date, it's not been studied how underlying network structure affect the point process's evolution, how robust they are to adversarial attacks and how they can become robust against adversarial attacks and non-stationary abrupt changes in data.

In Chapter 3, we study the dependency of Hawkes process virality on the network's structure and characteristics. Specifically, the results show that Hawkes process on networks with the same degree distribution and assortativity can exhibit very different levels of virality. With the inclusion of graphlets in predictive models of virality, R^2 values improve by 10-20% depending on the network and specific process. These results have implications for prediction of real network viral processes, as current methods are generally based upon degree distribution alone. Typically, approximate models of viral processes on networks can be somewhat readily written in terms of degree distribution, which explains the popularity of this basic network measure in understanding these processes. Better approximate models can be formulated by adding assortativity, which may also be plausible from an analytic point of view. Our results show that adding graphlet frequency distribution to these approximate models could be quite valuable. Here it may be beneficial to consider local graphlet statistics [122] around the source node of the process. For example, the local graphlet frequency around a Tweeter may help predict the number of re-shares of that user's Tweet.

In Chapter 4, we propose and study several white-box and black-box adversarial attacks against two state-of-the-art deep learning point processes that provide non-parametrical modeling of temporal point processes, and investigate the transferability of proposed adversarial attacks. Adversarial sample creation is a critical step, especially when the prediction from such models is utilized in safety and cost-critical applications. Moreover, we study the performance of the models facing non-stationary abrupt changes such as Covid-19. Finally, we examine how Hawkes process's parameters, in specific, are vulnerable to adversarial at-

tack. According to our experimental results, one can attack both predictive and parameter estimation performance of neural point processes with a small perturbation amount. Additionally, results imply the models are less sensitive to transferable attacks, still neural point processes are not robust to non-stationary changes in the data such as Covid-19. According to the results, although first-order white-box attacks are relatively analogous regarding predictive performances and underlying parameters, one crucial consideration is the associated cost of each attack.Iterative methods such as PGD, iFGSM, and miFGSM are expensive attacks in adversarial settings.

Finally, in Chapter 5, we study the robustness of previously studied state-of-the-art deep learning point processes that provide non-parametrical modeling of point processes. Specifically, we examined the effectiveness of adversarial training in improving robustness empirically. Adversarial training as a defense mechanism may be considered an attack-specific mechanism where first-order white-box attacks are the goal of the defense approach. In this setting, since previous results presented in Chapter 4 confirm PGD as a universal attack, the clean training set is augmented with adversarial samples from PGD. The associated cost is noteworthy regarding adversarial attacks and defense mechanisms. From an adversary viewpoint, low-cost attacks are preferred. However, from the defense standpoint, the ultimate goal of utilizing PGD in adversarial training as a defense mechanism is to increase the cost of adversarial attacks.

In addition to adversarial training, we propose general point process domain-adopted (GPDA) regularization as a low-cost attack-agnostic defense mechanism. This method leverages the theoretical property of the point process's compensator in a form of a regularization term. Experimental results suggest the proposed defense mechanisms can improve the robustness of both deep learning models against adversarial attacks imperially and boost their generalization on real data samples as shown by predictive performance and parametric modeling. The proposed GPDA defense mechanism joined by the exponential hazard model (EXP) may be considered as an alternative to complex and high cost modeling approaches as GPDA approach is able to improve the generalization and robustness of the Exponential hazard (EXP) model without any additional requirement or other complex function.

Finally, we propose a self-supervised adversarial detector that has been trained in the generative adversarial network (GAN) framework. According to the performance of pointGAN on adversarial detection, the proposed detector outperforms typical supervised and unsupervised adversarial detectors. Furthermore, the quality of generated sequences by pointGAN suggests examining GAN as a neural point process representation in which the generator can be used as a feature extractor.

In the following, we disclose how the implications of each chapter can be the direction to extend our methods. For instance, regarding the sensitivity analysis of point processes on the underlying network, future work may focus on analytical attempts at capturing the role graphlet frequency distribution plays in these viral processes now that their role has been experimentally verified. The statistically significant graphlets for a given network may also provide useful information for network optimization. When the goal is to reduce the spread of viruses, for example, to mitigate infectious virus breakouts such as Covid-19, one may wish to isolate nodes in a way that leads to the most significant reduction in viral spreading. While the degree may be used to make such selections, further gains may be made by considering assortativity and graphlet frequency.

Although the main contribution of Chapter 4 is adversarial attacks against temporal point processes, specifically the Hawkes process, multivariate Hawkes processes are an extension of our current framework. Here we don't consider the event marks, i.e., the associated information to each point. However, the proposed adversarial attacks are applicable to multivariate and marked temporal point processes.

Towards robustness of deep point processes, current work examines the effectiveness of adversarial training and a regularization approach. Nevertheless, alternative architectures such as transformers, graph neural networks, and Bayesian neural networks may be considered, regardless of their associated cost.

The primary focus of adversarial attacks and defense mechanisms here is l_∞ attacks. However, results on single-point adversarial attacks suggest the danger of l_2 localized adversarial attacks may be serious. On the other hand, the effectiveness of defense mechanisms against localized adversarial attacks suggests their capability in defending against corresponding l_2

norm adversarial attacks where the perturbation is localized. Therefore, exploring effect and robustness against l_2 norm attacks is the direction of future works.

In this research, we leveraged GAN for adversarial detection purposes. Nonetheless, based on the quality of generated sequences in addition to the quantitative performance of GAN frameworks in generating domain-specific text in our previous publication [65], another line of future research would be to use GAN-based representation and knowledge extraction tools for point processes and event sequence. Such embeddings may be helpful in crime linkage analysis, anomaly detection, and crime clustering.

In conclusion, in this research, we extend the topic of trustworthy machine learning to point processes and sequential data. Specifically, towards transparency, in Chapter 3, we exemplify how the virality of the Hawkes process can be explained by network characteristics such as graphlet distribution and assortativity. Furthermore, in Chapter 4 we step towards making neural point process models explainable and recognize their weaknesses and robustness by proposing several adversarial attacks against point processes in general and specific fields. Finally, regarding robustness against adversarial attacks, we utilize adversarial training and leverage theoretical approaches to propose a novel and flexible regularization method that improve the model robustness empirically. With the aid of our work in Chapter 3, one may wish to isolate nodes in a way that leads to the most significant reduction in viral spreading. For example, in the case of the dolphins-SIS simulation, choosing nodes that reduce the frequency of graphlets 21, 24, and 26 may provide a better mitigation strategy than other metrics like centrality or degree. Our proposed and studied adversarial attacks can be employed to identify weaknesses of point processes in sensitive fields such as criminology, natural disaster rescue systems, and stock market prediction. Also, our proposed GPDA method to improve the point process's robustness provides a low-cost defense mechanism that can be considered an alternative to computationally expensive deep learning modeling of point processes. Finally, the proposed pointGAN framework may be used in warning systems to detect adversarial sequences before providing the input sequence to the target network. Such a framework provides a robust and standalone system without requiring any specification on potential adversarial attacks.

REFERENCES

[1] S. Zhu, H. S. Yuchi, and Y. Xie, "Adversarial anomaly detection for marked spatio-temporal streaming data," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 8921–8925. DOI: 10.1109/ICASSP40776.2020.9053837.

[2] R. Shah, K. Muralidharan, and A. Parajuli, "Temporal point process models for nepal earthquake aftershocks," *International Journal of Statistics and Reliability Engineering*, vol. 7, no. 2, pp. 275–285, 2020.

[3] T. Murayama, S. Wakamiya, E. Aramaki, and R. Kobayashi, "Modeling and predicting fake news spreading on twitter," *arXiv preprint arXiv:2007.14059*, 2020.

[4] T. Murayama, S. Wakamiya, and E. Aramaki, "Fake news detection using temporal features extracted via point process," *arXiv preprint arXiv:2007.14013*, 2020.

[5] F. Adesola, A. Azeta, A. A. Oni, and F. Chidozie, "Forecasting violent crime hotspots using a theory-driven algorithm," *International Journal of Engineering Research and Technology (IJERT)*, 2019.

[6] C. Gilmour and D. J. Higham, "Modelling burglary in chicago using a self-exciting point process with isotropic triggering," *European Journal of Applied Mathematics*, pp. 1–23, 2021.

[7] R. Crane and D. Sornette, "Robust dynamic classes revealed by measuring the response function of a social system," *Proceedings of the National Academy of Sciences*, vol. 105, no. 41, pp. 15 649–15 653, 2008.

[8] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec, "Seismic: A self-exciting point process model for predicting tweet popularity," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2015, pp. 1513–1522.

[9] J. Leskovec, L. A. Adamic, and B. A. Huberman, "The dynamics of viral marketing," *ACM Transactions on the Web (TWEB)*, vol. 1, no. 1, p. 5, 2007.

[10] N. Berger, C. Borgs, J. T. Chayes, and A. Saberi, "On the spread of viruses on the internet," in *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, Society for Industrial and Applied Mathematics, 2005, pp. 301–310.

[11] M. B. Short, G. O. Mohler, P. J. Brantingham, and G. E. Tita, "Gang rivalry dynamics via coupled point process networks," *Discrete and Continuous Dynamical Systems - Series B*, vol. 19, no. 5, pp. 1459–1477, 2014, ISSN: 1531-3492. DOI: 10.3934/dcdsb.2014.19.1459. [Online]. Available: http://aimsciences.org/journals/displayArticlesnew.jsp?paperID=9866.

[12] A. Ganesh, L. Massoulie, and D. Towsley, "The effect of network topology on the spread of epidemics," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 2, Mar. 2005, 1455–1466 vol. 2. DOI: 10.1109/INFCOM.2005.1498374.

[13] D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos, "Epidemic thresholds in real networks," *ACM Trans. Inf. Syst. Secur.*, vol. 10, no. 4, 1:1–1:26, Jan. 2008, ISSN: 1094-9224. DOI: 10.1145/1284680.1284681. [Online]. Available: http://doi.acm.org/10.1145/1284680.1284681.

[14] Y. L-X, D. M, and Y. X, "The impact of the network topology on the viral prevalence: A node-based approach.," *PLoS ONE*, vol. 10, no. 7, 2015. DOI: 10.1371/journal.pone.0134507.

[15] L. Lovasz, *Eigenvalues of graphs*, http://web.cs.elte.hu/~lovasz/eigenvals-x.pdf, 2007.

[16] K. Dietz, "Models for vector-borne parasitic diseases," in *Vito Volterra Symposium on Mathematical Models in Biology*, Springer, 1980, pp. 264–277.

[17] R. M. Anderson, R. M. May, and B. Anderson, *Infectious diseases of humans: dynamics and control*. Wiley Online Library, 1992, vol. 28.

[18] Z. Dezső and A.-L. Barabási, "Halting viruses in scale-free networks," *Physical Review E*, vol. 65, no. 5, p. 055 103, 2002.

[19] D. S. Callaway, M. E. Newman, S. H. Strogatz, and D. J. Watts, "Network robustness and fragility: Percolation on random graphs," *Physical review letters*, vol. 85, no. 25, p. 5468, 2000.

[20] D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos, "Epidemic thresholds in real networks," *ACM Transactions on Information and System Security (TISSEC)*, vol. 10, no. 4, p. 1, 2008.

[21] O. Givan, N. Schwartz, A. Cygelberg, and L. Stone, "Predicting epidemic thresholds on complex networks: Limitations of mean-field approaches," *Journal of theoretical biology*, vol. 288, pp. 21–28, 2011.

[22] P. Van Mieghem, H. Wang, X. Ge, S. Tang, and F. Kuipers, "Influence of assortativity and degree-preserving rewiring on the spectra of networks," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 76, no. 4, pp. 643–652, 2010.

[23] S. Jalan and A. Yadav, "Assortative and disassortative mixing investigated using the spectra of graphs," *Physical Review E*, vol. 91, no. 1, p. 012 813, 2015.

[24] J. Qu, S.-J. Wang, M. Jusup, and Z. Wang, "Effects of random rewiring on the degree correlation of scale-free networks," *Scientific reports*, vol. 5, 2015.

[25] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. [Online]. Available: http://networkrepository.com.

[26] S. Xiao, J. Yan, S. M. Chu, X. Yang, and H. Zha, *Modeling the intensity function of point process via recurrent neural networks*, 2017. arXiv: 1705.08982 [cs.LG].

[27] H. Mei and J. Eisner, "The neural hawkes process: A neurally self-modulating multivariate point process," *arXiv preprint arXiv:1612.09328*, 2016.

[28] T. Omi, N. Ueda, and K. Aihara, "Fully neural network based model for general temporal point processes," *CoRR*, vol. abs/1905.09690, 2019.

[29] K. Gupta, J.-C. Pesquet, B. Pesquet-Popescu, F. Kaakai, and F. Malliaros, "An adversarial attacker for neural networks in regression problems," in *IJCAI Workshop on Artificial Intelligence Safety (AI Safety)*, 2021.

[30] Y. Deng, X. Zheng, T. Zhang, C. Chen, G. Lou, and M. Kim, *An analysis of adversarial attacks and defenses on autonomous driving models*, 2020. DOI: 10.48550/ARXIV.2002.02175. [Online]. Available: https://arxiv.org/abs/2002.02175.

[31] M. E. Newman, "Assortative mixing in networks," *Physical review letters*, vol. 89, no. 20, p. 208 701, 2002.

[32] M. Rahman, M. A. Bhuiyan, and M. A. Hasan, "GRAFT: An efficient graphlet counting method for large graph analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 10, pp. 2466–2478, 2014. DOI: 10.1109/TKDE.2013.2297929. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2013.2297929.

[33] M. Rahman and M. Al Hasan, "Link prediction in dynamic networks using graphlet," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2016, pp. 394–409.

[34] M. Rahman, M. Bhuiyan, and M. A. Hasan, "GRAFT: an approximate graphlet counting algorithm for large graph analysis," in *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, 2012, pp. 1467–1471. DOI: 10.1145/2396761.2398454. [Online]. Available: http://doi.acm.org/10.1145/2396761.2398454.

[35] M. Rahman and M. A. Hasan, "Sampling triples from restricted networks using MCMC strategy," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, 2014, pp. 1519–1528. DOI: 10.1145/2661829.2662075. [Online]. Available: http://doi.acm.org/10.1145/2661829.2662075.

[36] M. Bhuiyan, M. Rahman, M. Rahman, and M. A. Hasan, "GUISE: uniform sampling of graphlets for large graph analysis," in *12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012*, 2012, pp. 91–100. DOI: 10.1109/ICDM.2012.87. [Online]. Available: http://dx.doi.org/10.1109/ICDM.2012.87.

[37] M. Rahman, M. A. Bhuiyan, M. Rahman, and M. A. Hasan, "GUISE: a uniform sampler for constructing frequency histogram of graphlets," *Knowledge and Information Systems*, vol. 38, no. 3, pp. 511–536, 2014, Invited: best papers of ICDM'12. DOI: 10.1007/s10115-013-0673-3. [Online]. Available: http://dx.doi.org/10.1007/s10115-013-0673-3.

[38] T. K. Saha and M. A. Hasan, "Finding network motifs using MCMC sampling," in *Complex Networks VI - Proceedings of the 6th Workshop on Complex Networks CompleNet 2015, New York City, USA, March 25-27, 2015*, 2015, pp. 13–24. DOI: 10.1007/978-3-319-16112-9_2. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-16112-9_2.

[39] C. Gkantsidis, M. Mihail, and E. Zegura, "The markov chain simulation method for generating connected power law random graphs," in *In Proc. 5th Workshop on Algorithm Engineering and Experiments (ALENEX). SIAM*, 2003.

[40] S. Maslov and K. Sneppen, "Specificity and stability in topology of protein networks," *Science*, vol. 296, no. 5569, pp. 910–913, 2002.

[41] M. E. J. Newman, "The structure and function of complex networks," *SIAM REVIEW*, vol. 45, pp. 167–256, 2003.

[42] S. Xiao, M. Farajtabar, X. Ye, J. Yan, L. Song, and H. Zha, "Wasserstein learning of deep generative point process models," *arXiv preprint arXiv:1705.08051*, 2017.

[43] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song, "Recurrent marked temporal point processes: Embedding event history to vector," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1555–1564.

[44] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[45] E. Lewis and G. Mohler, "A nonparametric em algorithm for multiscale hawkes processes," *Journal of Nonparametric Statistics*, vol. 1, no. 1, pp. 1–20, 2011.

[46] I. J. Goodfellow, J. Shlens, and C. Szegedy, *Explaining and harnessing adversarial examples*, 2015. arXiv: 1412.6572 [stat.ML].

[47] A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard, "The robustness of deep networks: A geometrical perspective," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 50–62, 2017.

[48] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, Oct. 2019, ISSN: 1941-0026. DOI: 10.1109/tevc.2019.2890858. [Online]. Available: http://dx.doi.org/10.1109/TEVC.2019.2890858.

[49] K. Alparslan, Y. Alparslan, and M. Burlick, *Adversarial attacks against neural networks in audio domain: Exploiting principal components*, 2021. arXiv: 2007.07001 [cs.LG].

[50] S. Khorshidi, M. Al Hasan, G. Mohler, and M. B. Short, "The role of graphlets in viral processes on networks," *Journal of Nonlinear Science*, pp. 1–16, 2018.

[51] M. Farajtabar, J. Yang, X. Ye, H. Xu, R. Trivedi, E. Khalil, S. Li, L. Song, and H. Zha, "Fake news mitigation via point process based intervention," in *International Conference on Machine Learning*, 2017, pp. 1097–1106.

[52] H. Mei, G. Qin, and J. Eisner, "Imputing missing events in continuous-time event streams," in *International Conference on Machine Learning*, PMLR, 2019, pp. 4475–4485.

[53] H. Mei and J. Eisner, "The neural hawkes process: A neurally self-modulating multivariate point process," *CoRR*, vol. abs/1612.09328, 2016. arXiv: 1612.09328. [Online]. Available: http://arxiv.org/abs/1612.09328.

[54] C. Villani, *Optimal transport: old and new*. Springer Science & Business Media, 2008, vol. 338.

[55] E. Levina and P. Bickel, "The earth mover's distance is the mallows distance: Some insights from statistics," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, IEEE, vol. 2, 2001, pp. 251–256.

[56] L. Kantorovitch, "On the translocation of masses," *Management science*, vol. 5, no. 1, pp. 1–4, 1958.

[57] A. Geiger, D. Liu, S. Alnegheimish, A. Cuesta-Infante, and K. Veeramachaneni, "Tadgan: Time series anomaly detection using generative adversarial networks," in *2020 IEEE International Conference on Big Data (Big Data)*, IEEE, 2020, pp. 33–43.

[58] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Adversarial attacks on deep neural networks for time series classification," *2019 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2019. DOI: 10.1109/ijcnn.2019.8851936. [Online]. Available: http://dx.doi.org/10.1109/IJCNN.2019.8851936.

[59] A. Kurakin, I. Goodfellow, and S. Bengio, *Adversarial machine learning at scale*, 2016. DOI: 10.48550/ARXIV.1611.01236. [Online]. Available: https://arxiv.org/abs/1611.01236.

[60] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, *Towards deep learning models resistant to adversarial attacks*, 2017. DOI: 10.48550/ARXIV.1706.06083. [Online]. Available: https://arxiv.org/abs/1706.06083.

[61] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[62] G. R. Mode and K. A. Hoque, *Adversarial examples in deep learning for multivariate time series regression*, 2020. arXiv: 2009.11911 [cs.LG].

[63] F. P. Schoenberg, "Introduction to point processes," *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

[64] Y. Lee, P. J. Laub, T. Taimre, H. Zhao, and J. Zhuang, *Exact simulation of extrinsic stress-release processes*, 2021. DOI: 10.48550/ARXIV.2106.14415. [Online]. Available: https://arxiv.org/abs/2106.14415.

[65] S. Khorshidi, J. G. Carter, and G. Mohler, "Repurposing recidivism models for forecasting police officer use of force," in *2020 IEEE International Conference on Big Data (Big Data)*, IEEE, 2020, pp. 3199–3203.

[66] M.-A. Rizoiu, Y. Lee, S. Mishra, and L. Xie, *A tutorial on hawkes processes for events in social media*, 2017. arXiv: 1708.06401 [stat.ML].

[67] D. R. Brillinger, P. M. Guttorp, and F. P. Schoenberg, "Point processes, temporal," *Wiley StatsRef: Statistics Reference Online*, 2014.

[68] T. Omi, N. Ueda, and K. Aihara, "Fully neural network based model for general temporal point processes," *CoRR*, vol. abs/1905.09690, 2019. arXiv: 1905.09690. [Online]. Available: http://arxiv.org/abs/1905.09690.

[69] S. Baluja and I. Fischer, *Adversarial transformation networks: Learning to generate adversarial examples*, 2017. arXiv: 1703.09387 [cs.NE].

[70] A. T. Nguyen and E. Raff, *Adversarial attacks, regression, and numerical stability regularization*, 2018. arXiv: 1812.02885 [cs.LG].

[71] K. Gupta, J.-C. Pesquet, B. Pesquet-Popescu, F. Kaakai, and F. D. Malliaros, "An adversarial attacker for neural networks in regression problems," in *AISafety@IJCAI*, 2021.

[72] M.-I. Nicolae, M. Sinn, M. N. Tran, B. Buesser, A. Rawat, M. Wistuba, V. Zantedeschi, N. Baracaldo, B. Chen, H. Ludwig, I. M. Molloy, and B. Edwards, *Adversarial robustness toolbox v1.0.0*, 2018. DOI: 10.48550/ARXIV.1807.01069. [Online]. Available: https://arxiv.org/abs/1807.01069.

[73] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, *Boosting adversarial attacks with momentum*, 2017. DOI: 10.48550/ARXIV.1710.06081. [Online]. Available: https://arxiv.org/abs/1710.06081.

[74] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, *Ensemble adversarial training: Attacks and defenses*, 2017. DOI: 10.48550/ARXIV.1705.07204. [Online]. Available: https://arxiv.org/abs/1705.07204.

[75] C. Guo and L. Zhang, "A novel multiresolution spatiotemporal saliency detection model and its applications in image and video compression," *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 185–198, 2010. DOI: 10.1109/TIP.2009.2030969.

[76] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, *The limitations of deep learning in adversarial settings*, 2015. DOI: 10.48550/ARXIV.1511.07528. [Online]. Available: https://arxiv.org/abs/1511.07528.

[77] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, *Intriguing properties of neural networks*, 2013. DOI: 10.48550/ARXIV.1312.6199. [Online]. Available: https://arxiv.org/abs/1312.6199.

[78] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, *The space of transferable adversarial examples*, 2017. DOI: 10.48550/ARXIV.1704.03453. [Online]. Available: https://arxiv.org/abs/1704.03453.

[79] J. Q. Candela, C. E. Rasmussen, F. H. Sinz, O. Bousquet, and B. Schölkopf, "Evaluating predictive uncertainty challenge," in *MLCW*, 2005.

[80] B. Lakshminarayanan, A. Pritzel, and C. Blundell, *Simple and scalable predictive uncertainty estimation using deep ensembles*, 2017. arXiv: 1612.01474 [stat.ML].

[81] Y. Ogata, "Statistical models for earthquake occurrences and residual analysis for point processes," *Journal of the American Statistical association*, vol. 83, no. 401, pp. 9–27, 1988.

[82] F. Gerhard and W. Gerstner, "Rescaling, thinning or complementing? on goodness-of-fit procedures for point process models and generalized linear models," *arXiv preprint arXiv:1011.4188*, 2010.

[83] E. N. Brown, R. Barbieri, V. Ventura, R. E. Kass, and L. M. Frank, "The time-rescaling theorem and its application to neural spike train data analysis," *Neural computation*, vol. 14, no. 2, pp. 325–346, 2002.

[84] P. A. Meyer, *Séminaire de probabilités IX: Université de Strasbourg*. Springer, 2007, vol. 465.

[85] E. N. Brown, R. Barbieri, V. Ventura, R. E. Kass, and L. M. Frank, "The time-rescaling theorem and its application to neural spike train data analysis," *Neural Computation*, vol. 14, pp. 325–346, 2002.

[86] P. A. W. Lewis and G. S. Shedler, "Simulation of nonhomogeneous poisson processes with log linear rate function," *Biometrika*, vol. 63, no. 3, pp. 501–505, 1976, ISSN: 00063444. [Online]. Available: http://www.jstor.org/stable/2335727.

[87] V. Filimonov and D. Sornette, *Apparent criticality and calibration issues in the hawkes self-excited point process model: Application to high-frequency financial data*, 2013. DOI: 10.48550/ARXIV.1308.6756. [Online]. Available: https://arxiv.org/abs/1308.6756.

[88] F. Lorenzen, "Analysis of order clustering using high frequency data: A point process approach," *Swiss Federal Institute of Technology Zurich (ETH Zurich). http://arno. uvt. nl/show. cgi*, 2012.

[89] V. Tjeng, K. Xiao, and R. Tedrake, *Evaluating robustness of neural networks with mixed integer programming*, 2017. DOI: 10.48550/ARXIV.1711.07356. [Online]. Available: https://arxiv.org/abs/1711.07356.

[90] R. Bunel, I. Turkaslan, P. H. S. Torr, P. Kohli, and M. P. Kumar, *A unified view of piecewise linear neural network verification*, 2017. DOI: 10.48550/ARXIV.1711.00455. [Online]. Available: https://arxiv.org/abs/1711.00455.

[91] E. Wong and J. Z. Kolter, *Provable defenses against adversarial examples via the convex outer adversarial polytope*, 2017. DOI: 10.48550/ARXIV.1711.00851. [Online]. Available: https://arxiv.org/abs/1711.00851.

[92] K. Dvijotham, S. Gowal, R. Stanforth, R. Arandjelovic, B. O'Donoghue, J. Uesato, and P. Kohli, *Training verified learners with learned verifiers*, 2018. DOI: 10.48550/ARXIV.1805.10265. [Online]. Available: https://arxiv.org/abs/1805.10265.

[93] S. Gowal, K. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. Mann, and P. Kohli, *On the effectiveness of interval bound propagation for training verifiably robust models*, 2018. DOI: 10.48550/ARXIV.1810.12715. [Online]. Available: https://arxiv.org/abs/1810.12715.

[94] K. Lee and A. P. Chandrakasan, "Understanding the energy vs. adversarial robustness trade-off in deep neural networks," *IEEE Open Journal of Circuits and Systems*, vol. 2, pp. 843–855, 2021. DOI: 10.1109/OJCAS.2021.3116244.

[95] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognition*, vol. 84, pp. 317–331, Dec. 2018. DOI: 10.1016/j.patcog.2018.07.023. [Online]. Available: https://doi.org/10.1016%2Fj.patcog.2018.07.023.

[96] D. Su, H. Zhang, H. Chen, J. Yi, P.-Y. Chen, and Y. Gao, *Is robustness the cost of accuracy? – a comprehensive study on the robustness of 18 deep image classification models*, 2018. DOI: 10.48550/ARXIV.1808.01688. [Online]. Available: https://arxiv.org/abs/1808.01688.

[97] D. Jakubovitz and R. Giryes, *Improving dnn robustness to adversarial attacks using jacobian regularization*, 2018. DOI: 10.48550/ARXIV.1803.08680. [Online]. Available: https://arxiv.org/abs/1803.08680.

[98] C. Song, K. He, L. Wang, and J. E. Hopcroft, *Improving the generalization of adversarial training with domain adaptation*, 2018. DOI: 10.48550/ARXIV.1810.00740. [Online]. Available: https://arxiv.org/abs/1810.00740.

[99] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, *Virtual adversarial training: A regularization method for supervised and semi-supervised learning*, 2017. DOI: 10.48550/ARXIV.1704.03976. [Online]. Available: https://arxiv.org/abs/1704.03976.

[100] T. Miyato, S.-i. Maeda, M. Koyama, K. Nakae, and S. Ishii, *Distributional smoothing with virtual adversarial training*, 2015. DOI: 10.48550/ARXIV.1507.00677. [Online]. Available: https://arxiv.org/abs/1507.00677.

[101] E. Wong and Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," in *International Conference on Machine Learning*, PMLR, 2018, pp. 5286–5295.

[102] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," *arXiv preprint arXiv:1704.01155*, 2017.

[103] I. Rosenberg, A. Shabtai, Y. Elovici, and L. Rokach, *Defense methods against adversarial examples for recurrent neural networks*, 2019. arXiv: 1901.09963 [cs.CR].

[104] L. Yu, W. Zhang, J. Wang, and Y. Yu, *Seqgan: Sequence generative adversarial nets with policy gradient*, 2016. arXiv: 1609.05473 [cs.LG].

[105] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, 2014. eprint: arXiv:1406.2661.

[106] D. Li, D. Chen, J. Goh, and S.-k. Ng, *Anomaly detection with generative adversarial networks for multivariate time series*, 2018. DOI: 10.48550/ARXIV.1809.04758. [Online]. Available: https://arxiv.org/abs/1809.04758.

[107] S. Zhu, H. S. Yuchi, M. Zhang, and Y. Xie, *Sequential adversarial anomaly detection for one-class event data*, 2021. arXiv: 1910.09161 [stat.ML].

[108] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, *Distillation as a defense to adversarial perturbations against deep neural networks*, 2015. DOI: 10.48550/ARXIV.1511.04508. [Online]. Available: https://arxiv.org/abs/1511.04508.

[109] T. Aven, "A theorem for determining the compensator of a counting process," *Scandinavian Journal of Statistics*, vol. 12, no. 1, pp. 69–72, 1985, ISSN: 03036898, 14679469. [Online]. Available: http://www.jstor.org/stable/4615974.

[110] F. Papangelou, "Integrability of expected increments of point processes and a related random change of scale," *Transactions of the American Mathematical Society*, vol. 165, pp. 483–506, 1972.

[111] P. Embrechts, T. Liniger, and L. Lin, "Multivariate hawkes processes: An application to financial data," *Journal of Applied Probability*, vol. 48, no. A, pp. 367–378, 2011.

[112] A. Karazeev. (Aug. 17, 2017). "Generative adversarial networks (GANs): Engine and applications," Stats and Bots.

[113] C. Sornsoontorn. (Jan. 28, 2017). "How do GANs intuitively work?" Hacker Noon, [Online]. Available: https://hackernoon.com/how-do-gans-intuitively-work-2dda07f247a1.

[114] A. Radford, L. Metz, and S. Chintala, *Unsupervised representation learning with deep convolutional generative adversarial networks*, 2015. arXiv: 1511.06434 [cs.LG].

[115] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, *Unrolled generative adversarial networks*, 2016. arXiv: 1611.02163 [cs.LG].

[116] M. Mathieu, C. Couprie, and Y. LeCun, *Deep multi-scale video prediction beyond mean square error*, 2015. arXiv: 1511.05440 [cs.LG].

[117] Z. Liu, J. Wang, and Z. Liang, *Catgan: Category-aware generative adversarial networks with hierarchical evolutionary learning for category text generation*, 2019. arXiv: 1911.06641 [cs.CL].

[118] S. Khorshidi, G. Mohler, and J. G. Carter, "Assessing gan-based approaches for generative modeling of crime text reports," in *2020 IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2020, pp. 1–6. DOI: 10.1109/ISI49825.2020.9280487.

[119] C. Esteban, S. L. Hyland, and G. Rätsch, *Real-valued (medical) time series generation with recurrent conditional gans*, 2017. DOI: 10.48550/ARXIV.1706.02633. [Online]. Available: https://arxiv.org/abs/1706.02633.

[120] H. Sohrab, *Basic Real Analysis*. Birkhäuser Boston, 2003, ISBN: 9780817642112. [Online]. Available: https://books.google.com/books?id=gBPI%5C_oYZoMMC.

[121] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien, *A closer look at memorization in deep networks*, 2017. DOI: 10.48550/ARXIV.1706.05394. [Online]. Available: https://arxiv.org/abs/1706.05394.

[122] V. Dave, N. Ahmed, and M. A. Hasan, "E-CLoG: Counting edge-centric local graphlets," in *Proceedings of the 2017 IEEE International Conference on Big Data (Big Data)*, ser. BIG DATA '17, IEEE Computer Society, 2017.

A. SUPPLEMENTARY MATERIALS FOR ADVERSARIAL ATTACKS AGAINST DEEP TEMPORAL POINT PROCESSES

This Chapter presents supplementary materials on the results of the proposed adversarial attacks against the deep point process models on the specified datasets.

In FigureA.1 we are assessing the goodness of fit for both of our models to the specified datasets. The time-rescaling theorem and ordinary thinned residual process, as residual analysis approaches, on point processes ensure the deep learning models fit our data properly. In all figures, the green line represents the exponential hazard model and blue line represents the fully neural network model. In all figures, the gray area represents the 95% confidence band of the corresponding Poisson process.

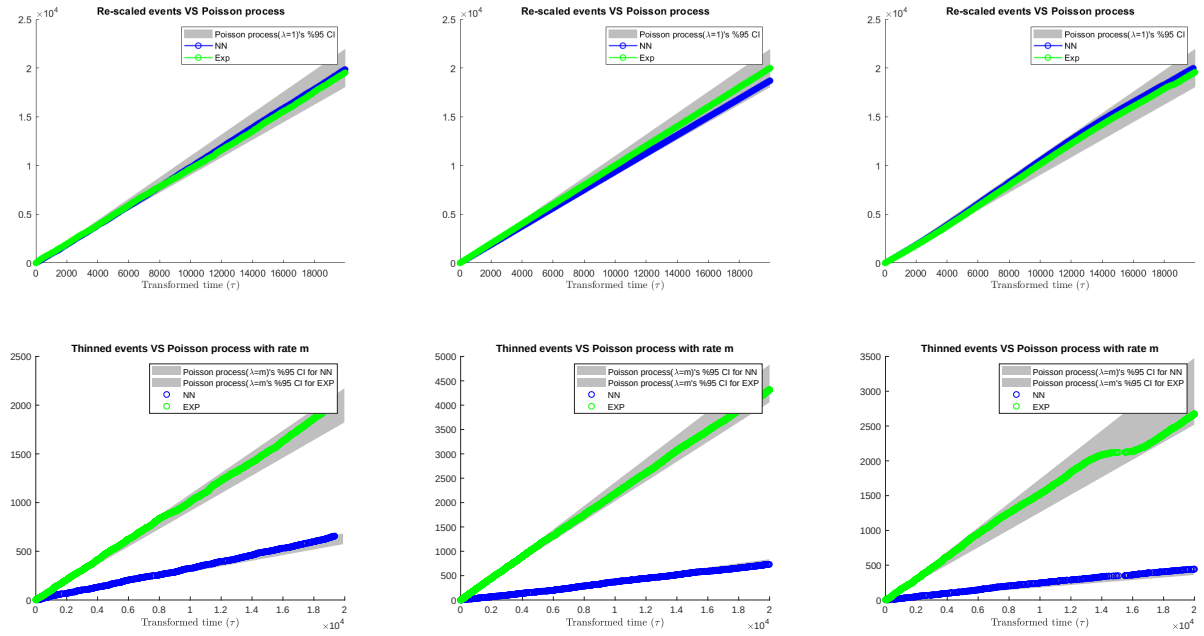


Figure A.1. Residual analysis and goodness of fit. Top: Re-scaled event times is withing unit rate Poisson process Down: Thinned data based on estimated λ matches Poisson process with rate m. Left: Hawkes process, Middle: Self-correcting, and Right: Non-stationary Poisson process.

In Table A.1, the complete version of the fully neural network model's performance on the specified datasets as well as adversarial attacks are presented. Consecutively, in Table

A.2, we present the performance of the exponential hazard model on the same datasets and adversarial attacks.

Table A.1. Predictive performance of fully neural network model against each attack on different point processes.

Data	Attack	**MNLL**(e − 01)	**MAE**(e − 01)	**FE** (e − 02)	**SMAPE** (e − 01)
Hawkes	No Attack	$4.1 \pm 2.838\text{e}-03$	$7.77 \pm 1.72\text{e}-04$	NA	NA
	PGD	$5.90 \pm 1.34\text{e}-02$	$8.09 \pm 1.15\text{e}-03$	$3.64 \pm 1.15\text{e}-03$	$3.13 \pm 5.35\text{e}-03$
	iFGSM	$6.54 \pm 1.56\text{e}-02$	$8.21 \pm 1.83\text{e}-03$	$4.75 \pm 1.65\text{e}-03$	$3.66 \pm 8.23\text{e}-03$
	R+FGSM	$6.04 \pm 8.67\text{e}-03$	$8.11 \pm 9.73\text{e}-04$	$3.83 \pm 7.68\text{e}-04$	$3.26 \pm 4.70\text{e}-03$
	miFGSM	$6.64E \pm 1.54\text{e}-02$	$8.20 \pm 1.43\text{e}-03$	$4.75 \pm 1.10\text{e}-03$	$3.67 \pm 5.35\text{e}-03$
	FGSM	$6.56 \pm 1.82\text{e}-02$	$8.18 \pm 2.09\text{e}-03$	$4.75 \pm 1.61\text{e}-03$	$3.60 \pm 5.67\text{e}-03$
	SM	$4.64 \pm 3.46\text{e}-03$	$7.91 \pm 6.23\text{e}-04$	$1.58 \pm 5.79\text{e}-04$	$1.56 \pm 3.61\text{e}-03$
	GD	$6.73 \pm 1.69\text{e}-02$	$7.85 \pm 1.45\text{e}-03$	$2.43 \pm 1.41\text{e}-03$	$3.33 \pm 8.04\text{e}-03$
	TM	$5.98 \pm 8.50\text{e}-03$	$8.36 \pm 1.10\text{e}-03$	$2.12 \pm 3.82\text{e}-03$	$7.08 \pm 4.51\text{e}-03$
Non Stationary Poisson	No Attack	$9.83 \pm 4.99\text{e}-03$	$10.00 \pm 4.09\text{e}-03$	NA	NA
	PGD	$10.25 \pm 5.812\text{e}-03$	$10.29 \pm 2.849\text{e}-03$	$3.38 \pm 4.24\text{e}-03$	$1.21 \pm 8.87\text{e}-03$
	iFGSM	$10.41 \pm 5.471\text{e}-03$	$10.38 \pm 5.75\text{e}-03$	$4.64 \pm 4.3\text{e}-03$	$1.57 \pm 7.63\text{e}-03$
	R+FGSM	$10.27 \pm 5.30\text{e}-03$	$10.29 \pm 3.62\text{e}-03$	$3.52 \pm 3.57\text{e}-03$	$1.27 \pm 1.05\text{e}-02$
	miFGSM	$10.44 \pm 1.05\text{e}-02$	$10.39 \pm 7.11\text{e}-03$	$4.60 \pm 4.34\text{e}-03$	$1.57 \pm 1.16\text{e}-02$
	FGSM	$10.41 \pm 4.88\text{e}-03$	$10.36 \pm 3.59\text{e}-03$	$4.53 \pm 3.64\text{e}-03$	$1.54 \pm 8.63\text{e}-03$
	SM	$9.89 \pm 5.46\text{e}-03$	$10.0 \pm 3.63\text{e}-03$	$4.37 \pm 7.81\text{e}-04$	$0.44 \pm 4.93\text{e}-03$
	GD	$9.86 \pm 5.96\text{e}-03$	$10.04 \pm 4.08\text{e}-03$	$4.30 \pm 6.42\text{e}-03$	$1.75 \pm 9.34\text{e}-03$
	TM	$12.91 \pm 2.76\text{e}-02$	$11.46 \pm 5.31\text{e}-03$	$40.56 \pm 3.66\text{e}-02$	$4.05 \pm 1.99\text{e}-02$
Self correcting	No Attack	$8.21 \pm 5.58\text{e}-02$	$4.97 \pm 6.80\text{e}-03$	NA	NA
	PGD	$14.04 \pm 1.16\text{e}-01$	$8.24 \pm 7.79\text{e}-02$	$36.13 \pm 8.67\text{e}-02$	$6.62 \pm 1.01\text{e}-01$
	iFGSM	$16.31 \pm 1.36\text{e}-01$	$9.52 \pm 9.94\text{e}-02$	$48.00 \pm 1.11\text{e}-01$	$7.77 \pm 1.05\text{e}-01$
	R+FGSM	$14.83 \pm 6.33\text{e}-02$	$8.70 \pm 4.95\text{e}-02$	$40.13 \pm 6.03\text{e}-02$	$7.08 \pm 6.44\text{e}-02$
	miFGSM	$16.99 \pm 1.78\text{e}-01$	$9.64 \pm 6.85\text{e}-02$	$49.53 \pm 7.55\text{e}-02$	$7.90 \pm 6.44\text{e}-02$
	FGSM	$16.58 \pm 1.86\text{e}-01$	$9.65 \pm 1.07\text{e}-01$	$49.29 \pm 1.18\text{e}-01$	$7.78 \pm 1.24\text{e}-01$
	SM	$9.02 \pm 6.66\text{e}-02$	$5.43 \pm 4.45\text{e}-03$	$5.69 \pm 3.34\text{e}-03$	$1.91 \pm 7.99\text{e}-03$
	GD	$15.49 \pm 1.72\text{e}-01$	$7.55 \pm 8.04\text{e}-02$	$31.47 \pm 9.86\text{e}-02$	$5.71 \pm 1.12\text{e}-01$
	TM	$11.94 \pm 7.39\text{e}-02$	$7.05 \pm 3.04\text{e}-02$	$49.79 \pm 4.84\text{e}-02$	$8.32 \pm 3.60\text{e}-02$

$*$ All numbers are the average over 10 times bootstrap experiments.

In Table A.3, Table A.4, Table A.5, and Table A.6, we present the transferability power of each adversarial attacks against both NN and EXP models. For Exponential hazard model (EXP), RNN units are the only parameters that can be modified to obtain transferred attacks.

Table A.2. Predictive performance of exponential hazard model against each attack.

Data	Attack	**MNLL**(e − 01)	**MAE**(e − 01)	**FE** (e − 02)	**SMAPE** (e − 01)
Hawkes	No Attack	$4.63 \pm 7.15e-03$	$8.01 \pm 1.26e-02$	NA	NA
	PGD	$5.38 \pm 9.29e-03$	$8.30 \pm 3.13e-02$	$2.62 \pm 4.36e-03$	$1.76 \pm 2.60e-02$
	iFGSM	$5.67 \pm 1.30e-02$	$8.26 \pm 3.33e-03$	$3.66 \pm 4.39e-03$	$2.32 \pm 2.36e-02$
	R+FGSM	$5.43 \pm 4.12e-03$	$8.19 \pm 1.59e-03$	$2.89 \pm 2.07e-03$	$1.93 \pm 1.27e-02$
	miFGSM	$5.65 \pm 1.32e-02$	$8.30 \pm 5.41e-03$	$3.57 \pm 5.27e-03$	$2.16 \pm 2.72e-02$
	FGSM	$5.72 \pm 8.93e-03$	$8.33 \pm 6.86e-03$	$4.01 \pm 3.20e-03$	$2.28 \pm 1.88e-02$
	SM	$4.76 \pm 7.46e-03$	$8.06 \pm 1.54e-02$	$0.67 \pm 5.69e-04$	$0.80 \pm 3.30e-03$
	GD	$5.28 \pm 1.59e-02$	$8.16 \pm 2.42e-02$	$4.73 \pm 1.76e-02$	$2.24 \pm 5.79e-03$
	TM	$6.67 \pm 3.70e-03$	$8.72 \pm 1.68e-02$	$28.58 \pm 3.52e-02$	$6.27 \pm 8.52e-03$
Non Stationary Poisson	No Attack	$9.7 \pm 1.97e-03$	$10.2 \pm 4.22e-03$	NA	NA
	PGD	$10.1 \pm 2.34e-03$	$10.5 \pm 2.96e-03$	$3.92 \pm 1.3e-03$	$1.29 \pm 2.91e-03$
	iFGSM	$10.3 \pm 3.741e-03$	$10.6 \pm 5.51e-03$	$2.24 \pm 2.12e-03$	$1.62 \pm 5.76e-03$
	R+FGSM	$10.19 \pm 1.59e-03$	$10.56 \pm 4.4e-03$	$4.24 \pm 1.29e-03$	$1.37 \pm 3.56e-03$
	miFGSM	$10.32 \pm 1.49e-03$	$10.60 \pm 3.52e-03$	$5.27 \pm 1.07e-03$	$1.65 \pm 3.50e-03$
	FGSM	$10.31 \pm 2.53e-03$	$10.62 \pm 4.07e-03$	$5.22 \pm 1.48e-03$	$1.62 \pm 4.31e-03$
	SM	$9.76 \pm 3.06e-03$	$10.29 \pm 6.5e-03$	$0.68 \pm 4.91e-04$	$0.46 \pm 2.18e-03$
	GD	$9.71 \pm 1.96e-03$	$10.24 \pm 4.68e-03$	$4.71 \pm 2.71e-03$	$1.83 \pm 5.52e-03$
	TM	$13.36 \pm 1.2e-02$	$11.42 \pm 1.5e-03$	$37.74 \pm 9.02e-02$	$4.25 \pm 4e-03$
Self correcting	No Attack	$7.82 \pm 9.04e-04$	$4.94 \pm 6.90e-04$	NA	NA
	PGD	$15.30 \pm 3.02e-02$	$8.52 \pm 4.76e-03$	$39.48 \pm 4.46e-03$	$7.05 \pm 4.58e-03$
	iFGSM	$18.86 \pm 2.72e-02$	$9.65 \pm 8.27e-03$	$50.41 \pm 8.51e-03$	$8.01 \pm 8.02e-03$
	R+FGSM	$15.87 \pm 2.49e-02$	$8.78 \pm 5.84e-03$	$41.83 \pm 5.89e-03$	$7.24 \pm 4.95e-03$
	miFGSM	$19.00 \pm 3.72e-02$	$9.65 \pm 1.36e-02$	$50.40 \pm 1.36e-02$	$8.02 \pm 1.17e-02$
	FGSM	$19.06 \pm 4.85e-02$	$9.60 \pm 9.67e-03$	$49.90 \pm 9.93e-03$	$7.98 \pm 9.04e-03$
	SM	$8.52 \pm 1.48e-03$	$5.37 \pm 3.78e-04$	$5.67 \pm 7.95e-04$	$1.92 \pm 2.64e-03$
	GD	$23.39 \pm 2.84e-01$	$7.05 \pm 6.79e-03$	$27.58 \pm 8.00e-03$	$5.74 \pm 1.20e-02$
	TM	$11.49 \pm 5.45e-03$	$6.91 \pm 5.31e-03$	$49.02 \pm 6.42e-03$	$8.36 \pm 4.68e-03$

∗ All numbers are the mean over 10 times bootstrap experiments.

Table A.3. MNLL compares attacks in transferred settings versus white-box settings against both fully neural network (NN) and exponential hazard (EXP). Stronger attack is marked in **bold**.

Attack	**Model**					
	NN				**EXP**	
		Transferred				**Transferred**
	White-box	**RNN** (256)	**NN** (256)	**Layers** (4)	**White-box**	**RNN** (256)
PGD	5.90e − 01	**6.58e-01**	4.73e − 01	5.58e − 01	5.38e − 01	**5.87e-01**
iFGSM	**6.54e-01**	5.64e − 01	5.50e − 01	**6.67e-01**	**5.67e-01**	5.48e − 01
R+FGSM	6.04e − 01	5.49e − 01	5.80e − 01	**6.59e-01**	5.43e − 01	**5.84e-01**
miFGSM	6.64e − 01	6.17e − 01	6.51e − 01	**6.62e-01**	5.65e − 01	**6.05e-01**
FGSM	6.56e − 01	5.23e − 01	**6.63e-01**	5.74e − 01	5.72e − 01	**4.75e-01**
SM	4.64e − 01	**6.57e-01**	6.53e − 01	5.59e − 01	4.76e − 01	**5.27e-01**
GD	**6.73e-01**	5.34e − 01	6.58e − 01	5.84e − 01	**5.28e-01**	4.63e − 01

∗ All numbers are averaged over 10 times bootstrap experiments.

Table A.4. MAE compares attacks in transferred settings versus white-box settings against both fully neural network (NN) and exponential hazard (EXP). Stronger attack is marked in **bold**.

Attack	**Model**					
	NN				**EXP**	
		Transferred				**Transferred**
	White-box	**RNN** (256)	**NN** (256)	**Layers** (4)	**White-box**	**RNN** (256)
PGD	8.09e − 01	**8.21e-01**	8.07e − 01	8.20e − 01	**8.30e-01**	8.11e − 01
iFGSM	8.21e − 01	**8.24e-01**	8.21e − 01	8.20e − 01	8.26e − 01	**8.30e-01**
R+FGSM	8.11e − 01	**8.25e-01**	7.85e − 01	8.16e − 01	**8.19e-01**	8.09e − 01
miFGSM	8.20e − 01	8.14e − 01	8.21e − 01	**8.21e-01**	**8.30e-01**	8.11e − 01
FGSM	8.18e − 01	7.84e − 01	8.19e − 01	**8.44e-01**	**8.33e-01**	8.14e − 01
SM	7.91e − 01	**8.19e-01**	7.84e − 01	8.18e − 01	8.06e − 01	**8.17e-01**
GD	7.85e − 01	8.11e − 01	**8.20e-01**	8.08e − 01	**8.16e-01**	7.89e − 01

∗ All numbers are the mean over 10 times bootstrap experiments.

Table A.5. FE compares attacks in transferred settings versus white-box settings against both fully neural network (NN) and exponential hazard (EXP). Stronger attacks is marked in **bold**.

Attack	**Model**					
	NN				**EXP**	
		Transferred				**Transferred**
	White-box	**RNN** (256)	**NN** (256)	**Layers** (4)	**White-box**	**RNN** (256)
PGD	3.64e − 02	**4.78e-02**	3.86e − 02	3.26e − 02	2.62e − 02	**3.53e-02**
iFGSM	4.75e − 02	3.51e − 02	3.13e − 02	**4.81e-02**	**3.66e-02**	3.05e − 02
R+FGSM	3.83e − 02	3.06e − 02	2.30e − 02	**4.71e-02**	2.89e − 02	**3.55e-02**
miFGSM	**4.75e-02**	4.62e − 02	4.69e − 02	4.68e − 02	3.57e − 02	**3.80e-02**
FGSM	4.75e − 02	2.13e − 02	**4.75e-02**	4.30e − 02	**4.01e-02**	5.21e − 03
SM	1.58e − 02	**4.74e-02**	2.71e − 02	3.29e − 02	6.73e − 03	**2.28e-02**
GD	2.43e − 02	4.22e − 02	**4.82e-02**	3.66e − 02	**4.73e-02**	1.54e − 02

∗ All numbers are the mean over 10 times bootstrap experiments.

Table A.6. SMAPE compares attacks in transferred settings versus white-box settings against both fully neural network (NN) and exponential hazard (EXP). Stronger attack is marked in **bold**.

Attack	**Model**					
	NN				**EXP**	
		Transferred				**Transferred**
	White-box	**RNN** (256)	**NN** (256)	**Layers** (4)	**White-box**	**RNN** (256)
PGD	3.13e − 01	**3.64e-01**	2.29e − 01	2.23e − 01	1.76e − 01	**2.79e-01**
iFGSM	3.66e − 01	2.34e − 01	2.19e − 01	**3.69e-01**	**2.32e-01**	1.96e − 01
R+FGSM	3.26e − 01	2.16e − 01	3.55e − 01	**3.60e-01**	1.93e − 01	**3.04e-01**
miFGSM	3.67e − 01	3.51e − 01	3.68e − 01	**3.70e-01**	2.16e − 01	**3.24e-01**
FGSM	3.60e − 01	3.32e − 01	**3.64e-01**	2.34e − 01	**2.28e-01**	7.99e − 02
SM	1.56e − 01	3.55e − 01	**3.69e-01**	2.19e − 01	8.05e − 02	**1.62e-01**
GD	3.33e − 01	2.30e − 01	**3.68e-01**	3.12e − 01	**2.24e-01**	1.52e − 01

∗ All numbers are averaged over 10 times bootstrap experiments.

B. SUPPLEMENTARY MATERIALS FOR ADVERSARIAL ROBUSTNESS

This Chapter presents supplementary materials on the results of the proposed defense mechanisms for deep point process models.

In Table B.1, the complete version of the pointGAN's performance in detecting adversarial samples against fully neural network (NN) model, compared to SVM and KNN are presented. Consecutively, in Table B.2, we present the performance of pointGAN's performance in detecting adversarial samples against the exponential hazard (EXP) model.

Finally, in Tables B.3 and B.4 the affect of adversarial attacks on the Hawkes process intensity function's parameters at prediction is presented.

Table B.1. Point-GAN attack detection performance on different adversarial attacks against the NN model

Attack	**Model**	**Accuracy**(%)	**Precision**	**Recall**	**F1 score**
PGD	pointGAN	55.99	0.98	0.89	0.62
	KNN	75.05	0.92	0.55	0.69
	SVM	50.70	0.60	0.05	0.10
iFGSM	pointGAN	56.26	0.98	0.89	0.62
	KNN	73.80	0.89	0.54	0.67
	SVM	50.68	0.57	0.05	0.09
R+FGSM	pointGAN	55.95	0.98	0.90	0.62
	KNN	73.90	0.89	0.53	0.67
	SVM	48.65	0.49	0.98	0.65
miFGSM	pointGAN	56.01	0.98	0.90	0.62
	KNN	74.05	0.90	0.54	0.67
	SVM	50.53	0.53	0.06	0.12
FGSM	pointGAN	55.92	0.98	0.89	0.61
	KNN	73.72	0.91	0.54	0.67
	SVM	49.50	0.60	0.02	0.04
SM	pointGAN	56.08	0.99	0.89	0.61
	KNN	73.32	0.89	0.52	0.66
	SVM	46.92	0.48	0.93	0.63
GD	pointGAN	56.30	0.98	0.90	0.62
	KNN	75.83	0.91	0.56	0.69
	SVM	47.80	0.48	0.96	0.64
TM	pointGAN	57.18	0.52	0.64	0.36
	KNN	69.07	0.81	0.52	0.63
	SVM	49.02	0.51	0.05	0.09

Table B.2. Point-GAN attack detection performance on different adversarial attacks against the EXP model

Attack	**Model**	**Accuracy**(%)	**Precision**	**Recall**	**F1 score**
PGD	pointGAN	56.18	0.98	0.90	0.62
	KNN	73.02	0.90	0.52	0.66
	SVM	50.15	0.60	0.03	0.06
iFGSM	pointGAN	55.93	0.98	0.90	0.62
	KNN	74.15	0.92	0.53	0.68
	SVM	49.65	0.55	0.02	0.03
R+FGSM	pointGAN	56.28	0.98	0.90	0.62
	KNN	74.07	0.89	0.54	0.67
	SVM	51.50	0.54	0.12	0.19
miFGSM	pointGAN	55.71	0.99	0.90	0.62
	KNN	73.55	0.87	0.53	0.66
	SVM	47.95	0.48	1.00	0.65
FGSM	pointGAN	55.99	0.98	0.89	0.61
	KNN	74.85	0.92	0.54	0.68
	SVM	47.00	0.48	0.84	0.61
SM	pointGAN	56.05	0.99	0.90	0.62
	KNN	73.45	0.87	0.54	0.67
	SVM	44.07	0.46	0.76	0.57
GD	pointGAN	56.33	0.98	0.89	0.61
	KNN	79.35	0.93	0.63	0.75
	SVM	45.77	0.47	0.90	0.62
TM	pointGAN	55.78	0.98	0.89	0.61
	KNN	69.07	0.81	0.52	0.63
	SVM	49.02	0.51	0.05	0.09

Table B.3. The effect of adversarial attacks against fully neural network (NN) model on Hawkes process parameters, μ, α, and β.

	Defense		**Income data**			**Predicted data**		
	GPDA	**AT**	$\boldsymbol{\mu}(\mathrm{e}-01)$	$\boldsymbol{\alpha}(\mathrm{e}+00)$	$\boldsymbol{\beta}(\mathrm{e}+00)$	$\boldsymbol{\mu}(\mathrm{e}-02)$	$\boldsymbol{\alpha}(\mathrm{e}-01)$	$\boldsymbol{\beta}(\mathrm{e}-01)$
No attack	–	–	0.129	0.874	0.911	1.25	8.79	8.43
	✓	–	0.129	0.874	0.911	1.26	8.77	8.60
	–	✓	0.1292	0.874	0.911	1.30	8.73	9.01
	✓	✓	0.129	0.874	0.911	1.31	8.73	9.15
PGD	–	–	9.98	1.00	1.30	1.20	8.82	7.94
	✓	–	9.98	1.00	1.30	1.26	8.78	8.56
	–	✓	9.98	1.00	1.30	1.24	8.80	8.27
	✓	✓	9.98	1.00	1.30	1.31	8.72	9.20
iFGSM	–	–	9.98	1.00	1.30	1.17	8.84	7.77
	✓	–	9.98	1.00	1.30	1.27	8.76	8.72
	–	✓	9.98	1.00	1.30	1.21	8.82	8.03
	✓	✓	9.98	1.00	1.30	1.31	8.73	9.13
R+FGSM	–	–	9.98	1.00	1.30	1.18	8.83	7.84
	✓	–	9.98	1.00	1.30	1.26	8.78	8.59
	–	✓	9.98	1.00	1.30	1.23	8.80	8.19
	✓	✓	9.98	1.00	1.30	1.31	8.72	9.17
miFGSM	–	–	9.98	1.00	1.30	1.18	8.84	7.80
	✓	–	9.98	1.00	1.30	1.27	8.77	8.63
	–	✓	9.98	1.00	1.30	1.24	8.79	8.32
	✓	✓	9.98	1.00	1.30	1.31	8.73	9.14
FGSM	–	–	9.98	1.00	1.30	1.18	8.83	7.83
	✓	–	9.98	1.00	1.30	1.26	8.78	8.58
	–	✓	9.98	1.00	1.30	1.21	8.81	8.06
	✓	✓	9.98	1.00	1.30	1.30	8.73	9.09
SM	–	–	9.98	1.00	1.30	1.23	8.81	8.19
	✓	–	9.98	1.00	1.30	1.26	8.77	8.60
	–	✓	9.98	1.00	1.30	1.29	8.75	8.86
	✓	✓	9.98	1.00	1.30	1.31	8.72	9.15
GD	–	–	9.96	0.99	0.908	1.12	8.88	7.37
	✓	–	9.96	1.00	0.80	1.24	8.80	8.29
	–	✓	9.97	1.00	1.03	1.20	8.83	7.99
	✓	✓	9.97	1.00	1.02	1.28	8.76	8.82
TM	–	–	9.96	0.999	1.15	2.73	7.30	53.5
	✓	–	9.96	1.00	1.07	1.79	8.22	25.61
	–	✓	9.97	1.00	1.15	2.92	7.12	72.14
	✓	✓	9.97	1.00	1.18	2.30	7.72	44.33

∗ All reported numbers are the mean over 10 bootstrapped experiments.
∗ $\frac{\alpha}{\beta}$ is the branching ratio of Hawkes process.

Table B.4. The effect of adversarial attacks against exponential hazard (EXP) model on Hawkes process parameters, μ, α, and β.

	Defense		**Income data**			**Predicted data**		
	GPDA	**AT**	$\boldsymbol{\mu}(\mathrm{e}-01)$	$\boldsymbol{\alpha}(\mathrm{e}+00)$	$\boldsymbol{\beta}(\mathrm{e}+00)$	$\boldsymbol{\mu}(\mathrm{e}-02)$	$\boldsymbol{\alpha}(\mathrm{e}-01)$	$\boldsymbol{\beta}(\mathrm{e}-01)$
No attack	–	–	0.129	0.874	0.911	1.32	8.70	9.43
	✓	–	0.129	0.874	0.911	1.33	8.70	9.48
PGD	–	–	9.98	1.00	1.30	1.32	8.71	9.19
	✓	–	9.98	1.00	1.30	1.31	8.71	9.31
iFGSM	–	–	9.98	1.00	1.30	1.32	8.71	9.16
	✓	–	9.98	1.00	1.30	1.32	8.70	9.48
R+FGSM	–	–	9.98	1.00	1.30	1.32	8.72	9.11
	✓	–	9.98	1.00	1.30	1.32	8.70	9.45
miFGSM	–	–	9.98	1.00	1.30	1.32	8.71	9.15
	✓	–	9.98	1.00	1.30	1.33	8.70	9.47
FGSM	–	–	9.98	1.00	1.30	1.32	8.71	9.24
	✓	–	9.98	1.00	1.30	1.33	8.69	9.52
SM	–	–	9.98	1.00	1.30	1.32	8.70	9.35
	✓	–	9.98	1.00	1.30	1.32	8.70	9.43
GD	–	–	9.96	0.999	1.02	1.23	8.81	8.20
	✓	–	9.97	1.00	0.95	1.29	8.75	8.93
TM	–	–	9.96	0.999	1.10	1.54	8.41	10.00
	✓	–	9.97	1.00	1.28	3.04	7.01	100.00

∗ All reported numbers are the mean over 10 bootstrapped experiments.
∗ $\frac{\alpha}{\beta}$ is the branching ratio of Hawkes process.

VITA

Samira graduated from Khajeh Nasir Toosi University of Technology (KNTU), also known as K. N. Toosi University of Technology, Iran, and received her bachelor of science from the Department of Computer Engineering in 2014. In 2016, she got admitted to Bauhaus University in Germany's master of computer science program. She was a student at Bauhaus when she got admitted to the Ph.D. program at Indiana University-Purdue University, Indianapolis. In addition to her academic background, she has more than seven years of professional experience from software engineer to C.T.O in the industry.

PUBLICATIONS

1. S. Khorshidi, B. Wang, and G. Mohler, "Adversarial attacks against deep temporal point process.", 21st IEEE International Conference on Machine Learning and Applications (IEEE ICMLA'22), Under review.

2. S. Khorshidi, J. Carter, G. Mohler, et al. "Explaining Crime Diversity with Google Street View, " J Quant Criminol 37, 361–391 (2021). https://doi.org/10.1007/s10940-021-09500-1

3. S. Khorshidi, M. Al Hasan, G. Mohler, et al. "The Role of Graphlets in Viral Processes on Networks, " J Nonlinear Sci 30, 2309–2324 (2020). https://doi.org/10.1007/s00332-018-9465-y

4. S. Khorshidi, G. Mohler and J. G. Carter, "Assessing GAN-based approaches for generative modeling of crime text reports," 2020 IEEE International Conference on Intelligence and Security Informatics (ISI), 2020, pp. 1-6, doi: 10.1109/ISI49825.2020.9280487.

5. S. Khorshidi, J. G. Carter and G. Mohler, "Repurposing recidivism models for forecasting police officer use of force," 2020 IEEE International Conference on Big Data (Big Data), 2020, pp. 3199-3203, doi: 10.1109/BigData50022.2020.9378173.

6. B. Abbaschian, and S. Khorshidi "A review of hybrid recommender systems." Ad Alta: Journal of Interdisciplinary Research 7.2 (2017).