Towards Robust Multivariate Time-Series Forecasting: Adversarial Attacks and Defense Mechanisms

Linbo Liu*
Dept. of Mathematics, UCSD
USA
linbo@ucsd.edu

Youngsuk Park†
AWS AI Labs
USA
pyoungsu@amazon.com

Trong Nghia Hoang
AWS AI Labs
USA
tnhoang@amazon.com

Hilaf Hasson
AWS AI Labs
USA
hashilaf@amazon.com

Jun Huan
AWS AI Labs
USA
lukehuan@amazon.com

ABSTRACT

As deep learning models have gradually become the main workhorse of time series forecasting, the potential vulnerability under adversarial attacks to forecasting and decision system accordingly has emerged as a main issue in recent years. Albeit such behaviors and defense mechanisms started to be investigated for the univariate time series forecasting, there are still few studies regarding the multivariate forecasting which is often preferred due to its capacity to encode correlations between different time series. In this work, we study and design adversarial attack on multivariate probabilistic forecasting models, taking into consideration attack budget constraints and the correlation architecture between multiple time series. Specifically, we investigate a sparse indirect attack that hurts the prediction of an item (time series) by only attacking the history of a small number of other items to save attacking cost. In order to combat these attacks, we also develop two defense strategies. First, we adopt randomized smoothing to multivariate time series scenario and verify its effectiveness via empirical experiments. Second, we leverage a sparse attacker to enable end-to-end adversarial training that delivers robust probabilistic forecasters. Extensive experiments on real dataset confirm that our attack schemes are powerful and our defend algorithms are more effective compared with other baseline defense mechanisms.

KEYWORDS

Machine learning, Time-series, Forecasting, Robustness

ACM Reference Format:
Linbo Liu*, Youngsuk Park†, Trong Nghia Hoang, Hilaf Hasson, and Jun Huan. . Towards Robust Multivariate Time-Series Forecasting: Adversarial Attacks and Defense Mechanisms . In *Proceedings of 8th SIGKDD International Workshop on Mining and Learning from Time Series – Deep Forecasting: Models, Interpretability, and Applications (MileTS '22).* ACM, New York, NY, USA, 9 pages.

1 INTRODUCTION

Model robustness in time-series forecasting has been a long-standing issue. Applications include achieving more consistent predictions for climate change [32], financial market [2], stable down-stream decision systems like in retail [6], resource planning for cloud computing [35], and optimal control vehicles [23]. Time series data are known to contain measurement noises and developing forecasting models less sensitive to such noise is highly desirable. Most of previous robust methods [28, 29, 46] in time series focused on traditional statistical models like vector autoregressive models and ARIMA [7], and Exponential Smoothing models [8], Prophet [45]. For those methods robustness refers to the capability of handling restricted notions of natural or non-adversarial noises, or model stability against outliers [11, 17]. In this paper we investigate a new type of model robustness for time-series forecasting on adversarial noises, which are new vulnerabilities that come along with deep learning, as elaborated next.

As deep learning-based forecastings [27, 34, 36, 38, 39, 47] increasingly replace traditional statistical counterparts, the potential issues of vulnerabilities against a distinct notion of adversarial perturbations [19, 43] has gained research attention in the time series modeling, which has only been explored mainly in the area of image classification. Therein, human-imperceptible adversarial perturbation can mislead classification outcomes of DNN models, triggering a vast amount of potential warnings to safety-critical systems such as self-driving cars [55]. Similar yet distinct speculative threat can occur in time series forecasting setting. Think of situation in cost-critical financial market where a financial institute makes profits based on its prediction of its client's stock price, which might be attacked by adversaries who want to degrade the financial institute's prediction. To make the attack imperceptible, the adversaries might devise a scheme that invests and hence changes the prices adversely for a small subset of stock indices, among a much larger pool of stock indices. Furthermore, the adversaries may not directly invest in the client's stock, which makes the attack harder to detect as there is no direct adverse investment into the target stock. This attack scheme is called indirect attack. See Figure 1 for a simulated example.

Such potential threats and simulation of a seemingly plausible and imperceptible attack to multivariate time-series forecasting (via an example of stock price prediction) are not straight-forward to be formulated through the existing framework developed in classification settings. Note several differences of forecasting against classification in terms of unique characteristic of time series, e.g.,

MileTS '22, Aug 15, 2022, Washington, DC
.

* Work done during the author's internship at AWS AI Labs. †Correspondence to: Youngsuk Park <pyoungsu@amazon.com>.

time horizon, multiple items, and probabilistic predictions of forecasters, e.g., quantiles. These differences open up the question of how adversarial perturbations and robustness should be defined in time series setting more properly. Extending recent endeavors on adversarial attack and robustness for univariate forecasting cases [14, 53], we formally investigate those in richer class of multivariate forecasting. Unlike an univariate forecasting class, new regime of sparse and indirect attack can be dominant and more effective under multivariate modelling, as mentioned in the previous stock example. This can not be captured from existing attack or defense designed from univariate forecasting class, especially when multiple time series are strongly correlated to each other. More formally, we try to tackle the following challenges:

1. Indirect Attack: Can we mislead the forecasting of a target time series via adverse perturbations made to others?

2. Attack Imperceptibility: Can the set of attacked time series be made sparse and nondeterministic to be less perceptible?

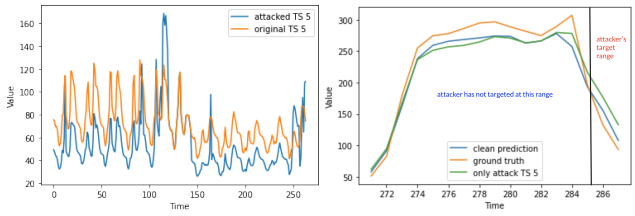
3. Robust Defense: If the above threats can be realized, can we devise a defense mechanism against them?

To address the above questions, we mainly propose three technical contributions listed below.

1. First, we devise a deterministic attack and show that adverse perturbations made to a subset of time series (not including the target time series) as described above can significantly alter the prediction outcome of the model. To be specific, we develop our deterministic attack (Section 3.2) based on the DeepVAR [38] model, which currently provides SOTA result to forecasting. Our attack is formulated as two-stage optimization task. The first phase finds an additive perturbation series to the authentic data such that DeepVAR's target statistics (e.g., prediction mean) is maximally altered in expectation, within a space of low-energy (hence, supposedly imperceptible) attacks. The second phase is then posed as a heuristic packing problem where all but k (with k being a user-specified parameter) rows of the perturbation matrix are zeroed out such that minimal amount of attack effect is lost.

2. Next, we develop probabilistic attack that learns to strategically make adverse perturbation to different (small) subsets of time series, making the attack much more stealth and harder to detect. Specifically, we formulate a probabilistic attack (Section 3.3) that relaxes the above k-hot constraint into a softer version that only requires the expectation of the attack vector rather than itself to be k-hot. Under such relaxation, we found that there is a provably approach to construct a learnable distribution over such k-hot attack space, with differentiable parameterization. This allows for the probabilistic attack model to elegantly merge the two separate phases of the deterministic attack. It can be shown empirically that an attack structured this way is often more effective (Section 5).

3. Finally, we propose two defense mechanisms. On the one hand, we adopt randomized smoothing technique [10, 25] to our setting. On the other hand, we devise a defense mechanism (Section 4.2) based on the differentiable formulation of the probabilistic attack



(a) (b)

Figure 1: Plots of (a) authentic (orange) and perturbed (blue) versions of time-series (TS) 5, which is selected by an attacker to mount an indirect attack on TS 1; and (b) ground-truth (orange), no-attack (blue) and under-attack (green) predictions for TS 1. No alteration was made to TS 1 but the value of TS 1 at the attack time step ($t = 288$) were adversely altered in the under-attack (green) setting, which can set the prediction of TS 1 significantly away from the ground truth.

above. Our defense is generated as the optimal solution to a minimax optimization task which minimizes the maximum expected damage caused by the probabilistic attacker that continually updates the generation of its adverse perturbation in response to the model updates. We also show the non-trivial effectiveness of our proposed defense against the aforementioned attacks (Section 5).

2 RELATED WORK AND BACKGROUND

In this section, we review some related work regarding time series forecasting, adversarial attacks and model robustness.

Deep Forecasting Models. Time series forecasting is a topic with long history and various models had been proposed from the previous century. Classic multivariate time series (MTS) models include moving average (MA), vector autoregressive (VAR), Autoregressive Integrated Moving Average (ARIMA). See Brockwell and Davis [7] for a comprehensive overview of classic models. However, traditional statistical tools become inaccurate when it comes to large collection of dataset like demand forecasting of large retailers, which motivates the usage of DNN-based forecasting models with more complex architecture to fit large scale dataset. The idea of applying neural network to time series forecasting dates back to Hu and Root [21] and stayed relatively quiet for a few decades. Recently, with the growth of large dataset and improvement of computing resources, more DNN-based forecasting models are investigated. Given the temporal dependency of time series data, RNN-based architectures have been proved a success for time series forecasting tasks [27, 36, 39, 47]. CNN-based forecasting models are also thoroughly studied in Bai et al. [5], Oord et al. [33], etc. In order to model the uncertainty, various probabilistic models have been proposed. For example, de Bézenac et al. [15], Rangapuram et al. [36], Salinas et al. [39] modeled the distribution of time series by neural networks and delivered probabilistic forecasters. Quantile-based methods [16, 22, 34] were also developed for consistent distribution-free uncertainty quantification. In multivariate cases, Sen et al. [41] leveraged a global matrix factorization and a local temporal network to *think globally, act locally*. Salinas et al. [38] generalized DeepAR [39] to multivariate cases and employed low-rank Gaussian copula process to reduce problem complexity raised by high dimensionality. Last, we refer interested readers to Lim and Zohren

[26] for a survey on forecasting with deep learning.

Adversarial Attack. Despite its success in various tasks, deep neural network is especially vulnerable to adversarial attacks [43] in the sense that even imperceptible adversarial noise can lead to completely different prediction. In computer vision, many adversarial attack schemes have been proposed. See Goodfellow et al. [19], Madry et al. [30] for attacking image classifiers and Dai et al. [13] for attacking graph structured data. In the field of time series, there is much less literature and even so, most existing studies on adversarial robustness of MTS models [20, 31] are restricted to regression and classification settings, which do not extend straightforwardly to forecasting with auto-regressive structures. Alternatively, Yoon et al. [53] studied adversarial attacks to probabilistic forecasting models as well as their defense mechanisms. However, these studies are restricted to univariate setting and cannot be extended to MTS setting with new potential threat of stealth, indirect attack as mentioned above.

Adversarial Robustness and Certification. The existence of adversarial attack significantly harms the performance of deep neural network. Thus, an extensive body of work has been devoted to quantify model robustness and many quantification algorithms were proposed, among which are *Fast-Lin/Fast-Lip* [50] that recursively compute local Lipschitz constant of a neural network, *PROVEN* [49] that certifies robustness in a probabilistic approach and *DeepZ* [42] that is based on abstract interpretation. To enhance model robustness, adversarial training and robust training are two popular techniques. In adversarial training [30, 52], a neural network is trained on the adversarial examples instead of the original ones; As for robust training, one trains the model by simultaneously minimizing the loss and maximizing certified robustness [50, 51].

Recently, randomized smoothing has gained increasing popularity as to enhance model robustness, which was proposed by Cohen et al. [10], Li et al. [25] as a defense approach with certification guarantee in image classification and has several variants [9, 24, 40, 54]. In a most recent work, Yoon et al. [53] adopted randomized smoothing technique to univariate forecasting models and developed theory therein. However, to the best of our knowledge, there is no prior work applying randomized smoothing to multivariate probabilistic models. In this work, we adopt randomized smoothing to multivariate time series and show the effectiveness of this technique via extensive empirical experiments.

To summarize, although there are prior works discussing adversarial attack and robustness for univariate probabilistic forecasting models, they cannot be readily generalized to multivariate cases with the extra technical difficulty raised by high-dimensionality. We present both deterministic and probabilistic approaches that overcome the discreteness issue caused by the required sparsity structure of the attack vector. For defense, we adopt randomized smoothing technique in multivariate cases and shows the effectiveness via empirical experiments on real dataset. Moreover, the probabilistic attack approach allows us to train an attacker in an end-to-end fashion, which in turns can be helpful in developing another end-to-end defense mechanism.

3 ADVERSARIAL ATTACK STRATEGIES

This section provides a quick review of the SOTA probabilistic forecasting model DeepVAR (Section 3.1), and introduces two approaches for sparse, indirect attack. First, a deterministic approach is developed which optimizes for a deterministic set of time series to be altered adversely in order to attack a target time series. This is achieved via a two-stage optimization process (Section 3.2). Next, to make the attack even less perceptible, a second non-deterministic approach is developed to instead optimize for a distribution over such subset of time series, which (unlike the former approach) can be learned end-to-end in a single stage (Section 3.3).

Algorithm 1 Deterministic sparse attack algorithm

Input: d-dimensional time series $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T) \in \mathbb{R}^{d\times T}$. A probabilistic forecasting model $f_\theta(\mathbf{x})$ which outputs density $p_\theta(\mathbf{y}|\mathbf{x})$. Future time horizon to attack $H \subseteq [\tau]$. Target time series $I \subseteq [d]$. Attack budget η. Sparsity k. Number of iteration N.

Output: Sparse attack $\delta \in \mathbb{R}^{d\times T}$ with row sparsity k and $\delta_{I,*} = \mathbf{0}$.

1. Initialize $\delta = \mathbf{0} \in \mathbb{R}^{d\times T}$
2. Draw a predicted sample $\hat{\mathbf{y}}$ from $p_\theta(\mathbf{y}|\mathbf{x})$. Get adversarial target value $\mathbf{t} = \chi(c\hat{\mathbf{y}})$.

for iteration $= 1, \ldots, N$ **do**

3. Compute predicted distribution: $p_\theta(\mathbf{y}|\mathbf{x}+\delta)$.
4. Compute expected loss under targeted attack:

$$\ell = \sum_{h\in H, i\in I} \mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}+\delta)} (y_{i,T+h} - t_{i,T+h})^2.$$

5. Use any first order method to update δ so as to minimize ℓ.
6. Clip δ with threshold η: $\delta_{i,t} = \delta_{i,t} \min\{1, \eta/|\delta_{i,t}|\}$

end for

7. For each time series $i \notin I$, compute cumulative perturbation over all time: $c_i = \sum_{t=1}^{T} |\delta_{it}|$ and sort c_i in descending order: $c_{\pi(1)} \geq c_{\pi(2)} \geq \cdots \geq c_{\pi(d)}$.
8. Keep $\delta_{\pi(1),*}, \ldots, \delta_{\pi(k),*}$ and set $\delta_{\pi(k+1),*}, \ldots, \delta_{\pi(d),*} = 0$.
9. Output δ.

3.1 Probabilistic Forecasting Models

Suppose a T-step history of a d-dimensional multivariate time series (MTS) $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T \in \mathbb{R}^d$ are given. Let $x_{i,t} \in \mathbb{R}$ denote the observed value of i-th time series at time t. The forecasting task is to predict the values $\mathbf{x}_{T+1}, \mathbf{x}_{T+2}, \ldots, \mathbf{x}_{T+\tau}$ of the MTS τ-step into the future. The prediction is often based on the observed values of $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T \in \mathbb{R}^d$. A probabilistic forecasting model $p_\theta(\mathbf{y}|\mathbf{x})$ is often characterized as an auto-regressive function mapping from the observed input $\mathbf{x} \in \mathbb{R}^{d\times T}$ to a distribution over future target values $\mathbf{y} \in \mathbb{R}^{d\times\tau}$. Its parameterization θ is often associated with a DNN in SOTA probabilistic deep forecasting models. Here, for notational convenience, we define $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T) \in \mathbb{R}^{d\times T}$ and $\mathbf{y} = (\mathbf{x}_{T+1}, \mathbf{x}_{T+2} \ldots, \mathbf{x}_{T+\tau}) \in \mathbb{R}^{d\times\tau}$. Also, for a matrix A, let $A_{i,*}$ denote the i-th row of A. We define the element-wise maximum norm and Frobenius norm of A as $\|A\|_{\max} = \max_{i,j} |A_{ij}|$, $\|A\|_F = (\sum_{i,j} A_{ij}^2)^{1/2}$. For a specific form of θ, we refer interested

readers to the DeepVAR paper [38]. In what follows, it suffices to present the main ideas with the abstraction θ.

Sparse Adversarial Attack. We will now formally define an attack to the above forecasting model. In particular, suppose we are interested in the statistic $\chi(\mathbf{y}) \in \mathbb{R}^m$ that is a function of the random vector $\mathbf{y}$. To stage an attack, we let δ denote an adverse perturbation to the input $\mathbf{x}$ such that the Euclidean distance between the expected statistic $\mathbb{E}_{p_\theta(\mathbf{y}|\mathbf{x}+\delta)}[\chi(\mathbf{y})]$ and an adversarial target $\mathbf{t}(\mathbf{x}) \in \mathbb{R}^m$ is minimized. Here, $\mathbf{t}(\mathbf{x})$ is the desired target specified by the adversaries, which is radically different from the clean prediction $\mathbb{E}_{p_\theta(\mathbf{y}|\mathbf{x})}[\chi(\mathbf{y})]$. Thus, the optimal attack can be found via solving the following constrained minimization task:

$$\begin{aligned} \min_{\delta \in \mathbb{R}^{d\times T}} \quad & J(\delta) \quad \triangleq \quad \left\| \mathbb{E}_{p_\theta(\mathbf{y}|\mathbf{x}+\delta)}\left[\chi(\mathbf{y})\right] - \mathbf{t}(\mathbf{x}) \right\|_2^2 \\ \text{s.t.} \quad & \|\delta\|_{\max} \quad \leq \quad \eta\,, \end{aligned} \tag{3.1}$$

where η specifies the desired energy of the attack and the above expectation is over $\mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{x}+\delta)$. Often, η is selected to be small to make the attack less perceptible. In the above, suppose that we want to mislead the forecasting for a set of pre-specified time-series with indices $I \in [d]$, our interested statistic will then be $\chi(\mathbf{y}) \triangleq \chi(\mathbf{y}_{I,*})$ which abstractly defines the predictive statistic corresponding to the rows of $\mathbf{y}$ with indices in I.

Stealth Attack. We now define a stealth attack δ as a sparse matrix such that its row sparsity $s(\delta) = |\{i : \delta_{i,*} \neq \mathbf{0}\}| \leq k$ where k is the desired level of sparsity. Intuitively, this means a stealth attack is configured such that only a small subset of its row might be non-zero whereas the rest of it is zero. A small value of k would therefore make the attack even less perceptible. Furthermore, since the interested statistic $\chi(\mathbf{y}) = \chi(\mathbf{y}_{I,*})$ involves the indices of time series in I, setting $\delta_{I,*} = \mathbf{0}$ can make the attack more stealthy as there is no direct adverse alternation in the time series appearing in χ. Therefore, an optimal stealth attack can be found by adding these constraints to Eq. (3.1),

$$\begin{aligned} \min_{\delta \in \mathbb{R}^{d\times T}} \quad & J(\delta) \quad \triangleq \quad \left\| \mathbb{E}_{p_\theta(\mathbf{y}|\mathbf{x}+\delta)}\left[\chi(\mathbf{y})\right] - \mathbf{t}(\mathbf{x}) \right\|_2^2 \\ \text{s.t.} \quad & \|\delta\|_{\max} \quad \leq \quad \eta,\ \ s(\delta) \ \leq\ k,\ \ \delta_{I,*} \ =\ \mathbf{0}\,. \end{aligned} \tag{3.2}$$

This however results in an intractable optimization task in general, so we provide two approximations in the subsequent sections.

3.2 Deterministic Attack

We first present a deterministic approach to solving (3.2) approximately. Here, the difficulty in optimizing (3.2) is due to the intractable constraint $s(\delta) \leq k$. To sidestep this, we use projected gradient descent (PGD) to numerically update the values of δ,

$$\delta^{(t+1)} \quad = \quad \prod_{B_\infty(0,\eta)} \left(\delta^{(t)} - \nabla_\delta J\left(\delta^{(t)}\right) \right), \tag{3.3}$$

where $\prod_{B_\infty(0,\eta)}$ is the projection onto the ℓ_∞-norm ball $B_\infty(0,\eta)$ with a radius η centered around the origin. Note that $\nabla_\delta J(\delta)$ involves the computation of the gradient of an expectation which is too complex to be analytically integrated. To overcome this intractability, we adopt the re-parameterized sampling approach used in Dang-Nhu et al. [14] and Yoon et al. [53]. Suppose δ^* denote the converged value of δ following the iterative update in Eq. (3.3), we solve for its sparse approximation via

$$\begin{aligned} \hat{\delta} \quad = \quad & \arg\min_{\delta} \quad \|\delta - \delta^*\|_F \\ & \text{s.t.} \quad s(\delta) \ \leq\ k, \quad \delta_{I,*} \ =\ \mathbf{0}\,. \end{aligned} \tag{3.4}$$

It is straightforward to see that (3.4) can be solved analytically. Given δ^*, we compute the absolute row sum $c_i = \sum_{t=1}^T |\delta^*_{i,t}|$ for $i \in I^c$ and sort them in descending order $c_{\pi(1)} \geq \cdots \geq c_{\pi(d-1)}$. Rows from the top k index $\pi(1), \pi(2), \ldots, \pi(k)$ will be kept in $\hat{\delta}$ while the other will be zeroed out, as described in Algorithm 1.

3.3 Probabilistic Attack

In this subsection, we further remove the two-stage heuristic approximation in Section 3.2, which is non-differentiable and cannot be optimized end-to-end, making it unsuitable to be integrated into a differentiable defense mechanism as described later in Section 4.2. The key issues here are in fact the non-convex and non-differentiable constraint in (3.2) which disables differentiable optimization via gradient descent. To sidestep this, we instead view the sparse attack vector as a random vector drawn from a distribution with differentiable parameterization which can be learned via gradient updates.

The core issue here is how to configure such distribution whose support is guaranteed to be within the space of sparse vectors. To achieve this, we configure this distribution as a learnable combination of a normal standard and a Dirac density, whose samples can be interpreted as differentiable transformation of samples drawn from a parameter-free normal standard – see Lemma 3.1. As we can update the parameters of the transformation via its gradient, we can learn the attack distribution with sparse support – see Lemma 3.2. Key to this parameterization is the ability to sample from a combination between Dirac and Gaussian densities, which is substantiated via the construction of a sparse layer as detailed below.

Sparse Layer. A sparse layer is configured as a conditional distribution $q_\Theta(\delta|\mathbf{x})$ such that $\mathbb{E}[s(\delta)] \leq k$ and $\delta_{I,*} = \mathbf{0}$ where $\delta \sim q_\Theta(\delta|\mathbf{x})$, I is the set of target time-series which are not to be altered, and k is the user-specified level of sparsity as defined before. Let $\Theta = (\beta, \gamma)$. We treat each row $\delta_{i,*}$ of δ as an independent sample drawn from $q_i(\delta_{i,*}|\mathbf{x};\beta,\gamma)$ parameterized by β and γ, as defined below:

$$\begin{aligned} q_i\Big(\delta_{i,*}|\mathbf{x};\beta,\gamma\Big) \quad \triangleq \quad & r_i\left(\gamma\right) \cdot q'_i\Big(\delta_{i,*}|\mathbf{x};\beta\Big) \\ + \quad & \Big(1 - r_i(\gamma)\Big) \cdot D\Big(\delta_{i,*}\Big)\,, \end{aligned} \tag{3.5}$$

where $D(\delta_{i,*})$ is the Dirac density concentrated at $\delta_{i,*} = \mathbf{0}$ and $q'_i(\delta_{i,*}|\mathbf{x};\beta)$ is a Gaussian density whose mean and variance are functions of $\mathbf{x}$ which are parameterized by β that can be weights of a DNN. The combination weight $r_i(\gamma)$ on the other hand denotes the probability mass of the event $\delta_{i,*} = \mathbf{0}$, which is parameterized by γ. Intuitively, this means the choice of $\{r_i(\gamma)\}_{i=1}^n$ controls the row sparsity of the random matrix δ, which can be calibrated to enforce that $\mathbb{E}[s(\delta)] \leq k$. We will show in Lemma 3.1 how samples can be drawn from the combined density in (3.5). Then, Lemma 3.2

shows why sample δ drawn from (3.5) would meet the constraint $\mathbb{E}[s(\delta)] \leq k$. Put together, Lemma 3.1 and Lemma 3.2 enables differentiable optimization of a sparse attack distribution as desired.

Lemma 3.1. *Let $\delta'_{i,*} \sim q'_i(\cdot|\mathbf{x}, \beta)$ and $u_i \sim \mathcal{N}(0,1)$ for $i = 1, \ldots, d$. Define $\delta_{i,*} = \delta'_{i,*} * \mathbb{I}(u_i \leq \Phi^{-1}(r_i(\gamma)))$. Then, $\delta_{i,*} \sim q_i(\delta_{i,*}|\mathbf{x}; \beta, \gamma)$.*

Here, $q_i(\cdot|\mathbf{x}; \beta, \gamma)$ is given in (3.5) and Φ^{-1} is the inverse cumulative of the standard normal distribution.

Proof. We can compute

$$\begin{aligned} \mathbb{P}(\delta_{i,*} = \mathbf{0}) &= 1 - \mathbb{P}\left(u_i \leq \Phi^{-1}\left(r_i(\gamma)\right)\right) \\ &= 1 - r_i(\gamma) \end{aligned} \tag{3.6}$$

That is, with probability $1 - r_i(\gamma)$, $\delta_{i,*} = 0$. Equivalently, $\delta_{i,*}$ is distributed by a degenerated probability measure with Dirac density $D(\delta_{i,*})$ concentrated at 0. On the other hand, with probability $r_i(\gamma)$, $\delta_{i,*}$ is distributed as $q'_i(\cdot|\mathbf{x}; \beta)$. Combining the two cases, it follows that $\delta_{i,*}$ is distributed by a mixture of $q'_i(\cdot|\mathbf{x}; \beta)$ and $D(\delta_{i,*})$ with weights $r_i(\gamma)$ and $1 - r_i(\gamma)$ respectively. □

For implementation, observing that the second property $\delta_{I,*} = \mathbf{0}$ can always be satisfied by zeroing out the I rows of δ. Thus, for simplicity, we ignore this constraint. Let $q'_i(\cdot|\mathbf{x}; \beta)$ be dense distributions, e.g. $\mathcal{N}(\mu(\beta), \sigma^2(\beta)I)$, over $\mathbb{R}^T$ and $u_i \sim \mathcal{N}(0,1)$ for $i \in [d]$. We can construct a binary mask,

$$\text{mask}_i = \mathbb{I}\left(u_i \leq \Phi^{-1}\left(r_i(\gamma)\right)\right), \quad i \in [d]$$

where $r_i(\gamma) \triangleq \left(k\gamma_i^{1/2}/\sqrt{d}\right) / \left(\sum_{i=1}^d \gamma_i\right)^{1/2}$.

Next, for each $i \in [d]$, we draw $\delta'_{i,*}$ from $q'_i(\cdot|\mathbf{x}, \beta)$ and obtain $\delta_{i,*}$ by $\delta_{i,*} = \delta'_{i,*} * \text{mask}_i$. Finally, we set $\delta_{I,*} = \mathbf{0}$. Lemma 3.2 below verifies the required sparsity property in expectation, thus completing our differentiable sparse attack.

Lemma 3.2. *Let $\delta \sim q_\Theta(\cdot|\mathbf{x})$. Then, $\mathbb{E}[s(\delta)] \leq k$.*

Proof. By the construction of $r_i(\gamma)$,

$$\begin{aligned} \mathbb{E}\left[s(\delta)\right] &= \sum_{i=1}^d \mathbb{E}\left[\mathbb{I}\left(u_i \leq \Phi^{-1}\left(r_i(\gamma)\right)\right)\right] \\ &= \sum_{i=1}^d \mathbb{P}\left(u_i \leq \Phi^{-1}\left(r_i(\gamma)\right)\right) \\ &= \sum_{i=1}^d r_i(\gamma) = \frac{k}{\sqrt{d}} \cdot \frac{\sum_{i=1}^d \gamma_i^{1/2}}{\left(\sum_{i=1}^d \gamma_i\right)^{1/2}} \leq k \end{aligned}$$

□

Optimizing Sparse Layer. The differentiable parameterization of the above sparse layer can be optimized (for maximum attack impact) via minimizing the expected distance between the attacked statistic and adversarial target:

$$\min_{\Theta} \; \mathbb{E}_{\delta \sim q_\Theta(\cdot|\mathbf{x})} \left\| \mathbb{E}_{\mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{x}+\delta)}[\chi(\mathbf{y})] - \mathbf{t}(\mathbf{x}) \right\|_2^2, \tag{3.7}$$

This attack is probabilistic in two ways: First, the magnitude of the perturbation δ is a random variable from distribution $q(\cdot|\mathbf{x})$. Second, the non-zero components of the mask depend on the random

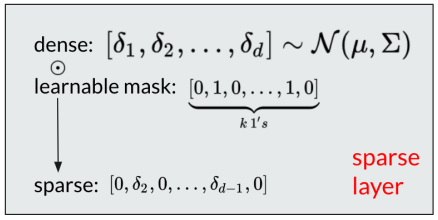


Figure 2: Work flow of sparse layer

Gaussian samples u_i in Lemma 3.1, which brings another degree of non-determinism into the design, making the attack more stealth and harder to detect.

Discussion. There are three important advantages of the above probabilistic sparse attack. First, by viewing the attack vector as random variable drawn from a learnable distribution instead of fixed parameter to be optimized, we are able to avoid solving the NP-hard problem (3.2) as usually approached in previous literature [12]. Second, our approach introduces multiple degree of non-determinism to the attack vector, apparently making it more stealth and powerful (see the experiments in Section 5). Last, as the attack model is entirely differentiable, it can be directly integrated as part of a differentiable defense mechanism that can be optimized via gradient descent in an end-to-end fashion – see Section 4.2 for more details.

4 DEFENSE AGAINST ADVERSARIAL ATTACK

The adversarial attack on probabilistic forecasting models was investigated in Dang-Nhu et al. [14], Yoon et al. [53] under univariate time series setting. Many efforts have been made to defend against adversarial attack. Data augmentation has been widely applied in forecasting [48] and can improve model robustness. In the following section, we go beyond data augmentation and introduce more advanced techniques to enhance model robustness via randomized smoothing [10] and mini-max defense using sparse layer.

4.1 Randomized Smoothing

Randomized smoothing was first proposed by Cohen et al. [10], Li et al. [25] in the field of computer vision and generalized to univariate time series setting by Yoon et al. [53]. Randomized smoothing is a post-training process and can be applied to any forecasting model $f_\theta(\mathbf{x})$, or $f(\mathbf{x})$ if the context is clear. Mathematically, let f be a random function that maps $\mathbf{x} \in \mathbb{R}^{d \times T}$ to a random vector $f(\mathbf{x})$ in $\mathbb{R}^d$ and denote the CDF of $f(\mathbf{x})$ as $F_{\mathbf{x}}(\mathbf{r}) = \mathbb{P}(f(\mathbf{x}) \leq \mathbf{r})$, where $\leq$ denotes element-wise inequality. Let $g_\sigma(\mathbf{x})$ be the randomized smoothing version of $f(\mathbf{x})$ with noise level σ and $g_\sigma(\mathbf{x})$ is also a random vector in $\mathbb{R}^d$ whose CDF is defined as

$$G_{\mathbf{x},\sigma}(\mathbf{r}) = \mathbb{E}_{\mathbf{z} \sim \mathcal{N}_{d \times T}(0, \sigma^2 I)} \left[\mathbb{P}\left(f(\mathbf{x}+\mathbf{z}) \leq \mathbf{r}\right)\right],$$

As shown by Yoon et al. [53], sample paths drawn from $G_{\mathbf{x},\sigma}(\mathbf{r})$ have robustness certification in univariate cases.

Implementation. To get n future samples from randomized smoothing forecaster, we independently draw n isotropic Gaussian noises $\epsilon_1, \ldots, \epsilon_n \sim \mathcal{N}_{d \times T}(0, \sigma^2 I)$ and compute the predicted distribution

$f_\theta(\mathbf{x}(1+\epsilon_i))$ for future time series. For each $f_\theta(\mathbf{x}(1+\epsilon_i))$, draw a sample $\hat{\mathbf{y}}^{(i)} \sim f_\theta(\mathbf{x}(1+\epsilon_i))$ and collect $\hat{\mathbf{y}}^{(\mathbf{i})}$ for $i = 1, \ldots, n$. These will be the sample paths under randomized smoothing.

4.2 Mini-max Defense

We notice that our sparse layer can not only be used as an attacker, but is also helpful as a defense procedure.

Formulation. We randomly initialize a sparse layer g_Θ with sparsity k as a hyper-parameter and a forecasting model f_θ from scratch. For each data point $\mathbf{x}$ in the training set, the sparse layer g_Θ is used to generate a sparse adversarial example $\hat{\mathbf{x}}$, which is then fed into f_θ to complete training phase. Specifically, in each epoch, the first step is to update the parameters of the sparse layer by maximizing the model's deviation from the true prediction

$$\ell_g = \sum_{i=1}^{n} \mathbb{E}_{\delta \sim g_\Theta(\mathbf{x}_i;k)} \|\mathbb{E}_{\mathbf{y}_i \sim f_\theta(\mathbf{x}_i+\delta)}[\mathbf{y}_i] - \mathbf{y}_i^{\text{true}}\|,$$

where y_i^{true} is the ground truth prediction. In the second step, we train the model f_θ on the corrupted examples generated by the current g_Θ. In other words, we update θ to maximize the model likelihood:

$$\ell_f = \sum_{i=1}^{n} \mathbb{E}_{\delta_i \sim g_\Theta(\mathbf{x}_i;k)} \log p_\theta(\mathbf{y}_i^{\text{true}} | \mathbf{x}_i + \delta_i).$$

Note that g_Θ and f_θ compete over one another in the sense that in each epoch, g_Θ is trained to generate effective attack that could harm f_θ and f_θ is then trained to defend the attack from g_Θ. We call this defense mechanism a mini-max defense. Similar ideas have been exploited in deep generative models, such as GAN [18] and WGAN [3]. See Algorithm 2 for a detailed description.

Algorithm 2 Mini-max defense algorithm

Input: Forecasting dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i^{\text{true}}\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^{d \times T}$ and $\mathbf{y}_i^{\text{true}} \in \mathbb{R}^{d \times \tau}$ is obtained from backtest window using historical data. Hyper-parameter k as the sparsity level in the sparse layer $g_\Theta(\mathbf{x}; k)$
Output: A forecasting model $f_\theta(\mathbf{x})$.
for epoch $= 1, \ldots, N$ **do**

1. Compute loss for the sparse layer $g_\Theta(\mathbf{x}; k)$:

$$\ell_g = -\sum_{i=1}^{n} \mathbb{E}_{\delta \sim g_\Theta(\mathbf{x}_i;k)} \|\mathbb{E}_{\mathbf{y}_i \sim f_\theta(\mathbf{x}_i+\delta)}[\mathbf{y}_i] - \mathbf{y}_i^{\text{true}}\|.$$

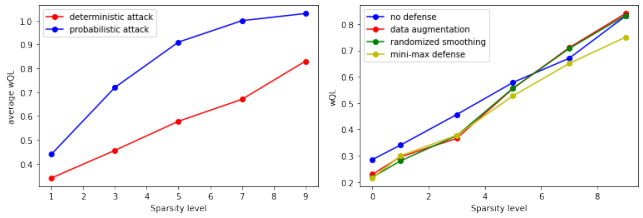
2. Update Θ in the sparse layer to minimize ℓ_g.
3. Let $p_\theta(\cdot|\mathbf{x})$ be the output distribution of $f_\theta(\mathbf{x})$. Compute likelihood for model $f_\theta(\mathbf{x})$:

$$\ell_f = \sum_{i=1}^{n} \mathbb{E}_{\delta_i \sim g_\Theta(\mathbf{x}_i;k)} \log p_\theta(\mathbf{y}_i^{\text{true}} | \mathbf{x}_i + \delta_i).$$

4. Update θ in forecasting model to maximize ℓ_f.

end for

Different from the sparse layer used in attack, this sparse layer in defense does not have access to the attack sparsity or the set of target time series I. Hence, we need to set the sparsity k as a hyper-parameter and skip the last step of the sparse layer described in Section 3.3 where we set $\delta_{I,*} = \mathbf{0}$.



(a) (b)

Figure 3: Plots of (a) averaged wQL under sparse indirect attack against the sparsity level on electricity dataset. The underlying model is a clean DeepVAR without defense. Target time series $I = \{1\}$ and attacked time stamp $H = \{\tau\}$; and (b) average wQL under different defense mechanisms on electricity dataset. The attack type is deterministic attack. Target time series $I = \{1\}$ and attacked time stamp $H = \{\tau\}$.

5 EXPERIMENTS

We conduct numerical experiments to demonstrate the effect of our proposed indirect sparse attack on a probabilistic DeepVAR model [38] and compare various defense mechanisms including data augmentation, randomized smoothing and mini-max defense. The experiments are performed on standard real datasets for time series forecasting including Taxi [44] and UCI Electricity [4] datasets preprocessed as in Salinas et al. [39].

5.1 Experiment Setups

In empirical experiments, we target the prediction of the first time series at the last prediction time step, i.e. target time series $I = \{1\}$ and time horizon to attack $H = \{\tau\}$, the last time step, so, $\chi(\mathbf{y}) = x_{1,T+\tau}$. For the adversarial target $\mathbf{t}(\mathbf{x})$, we first draw a prediction $\hat{\mathbf{x}}$ from un-attacked model $p_\theta(\cdot|\mathbf{x})$ and choose $\mathbf{t} = c_1 \hat{x}_{1,T+\tau}$ for some $c_1 > 0$. Note that c_1 should be away from 1 to reflect adversarial target. The attack energy $\eta = c_2$. When adding adversarial perturbation δ to $\mathbf{x}$, always add relative change $\mathbf{x}(1+\delta)$. Unless otherwise stated, the number of sample paths drawn from the prediction distribution $n = 100$ to quantify quantiles $q_{i,t}^{(\alpha)}$.

Metrics. We measure the performance of model under attacks by the popular metric especially for probabilistic forecasting models: weighted quantile loss (wQL), which is defined as

$$\text{wQL}(\alpha) = 2\frac{\sum_{i,t}[\alpha \max(x_{i,t} - q_{i,t}^{(\alpha)}, 0) + (1-\alpha)\max(q_{i,t}^{(\alpha)} - x_{i,t}, 0)]}{\sum_{i,t}|x_{i,t}|},$$

where $\alpha \in (0,1)$ is a quantile level. In practical application, under-prediction and over-prediction may cost differently, suggesting wQL should be one's main consideration especially for probabilistic forecasting models. In the subsequent sections, we calculate average wQL over a range of $\alpha = [0.1, 0.2, \ldots, 0.9]$ and evaluate the performance in terms of averaged wQL.

Data augmentation and randomized smoothing. Following the convention in Dang-Nhu et al. [14], Yoon et al. [53], we use relative noises in both data augmentation and randomized smoothing. That

Table 1: Average wQL on electricity dataset under deterministic attack. Target time series $I = \{1\}$ and attacked time stamp $H = \{\tau\}$. Smaller is better.

Sparsity	no defense	data augmentation	randomized smoothing	mini-max defense
no attack	0.2853±0.0825	0.2288±0.0792	0.2176±0.0700	**0.2154**±0.0705
1	0.3410±0.0946	0.2949±0.0716	**0.2826**±0.0718	0.2990±0.0772
3	0.4559±0.1344	**0.3655**±0.1097	0.3757±0.1012	0.3775±0.0923
5	0.5770±0.1772	0.5554±0.1636	0.5560±0.1751	**0.5273**±0.1558
7	0.6687±0.2131	0.7076±0.2321	0.7072±0.2308	**0.6506**±0.2111
9	0.8282±0.2847	0.8412±0.2896	0.8327±0.2786	**0.7503**±0.2588

Table 2: Metrics on electricity dataset under probabilistic attack using sparse layer. Target time series $I = \{1\}$ and attacked time stamp $H = \{\tau\}$. Smaller is better.

Sparsity	no defense	data augmentation	randomized smoothing	mini-max defense
no attack	0.2909±0.0748	0.2374±0.0764	**0.2237**±0.0750	0.2342±0.0710
1	**0.4364**±0.1296	0.5923±0.0913	0.5940±0.1142	0.4935±0.1450
3	0.7245±0.2434	0.5738±0.1759	**0.4581**±0.1301	0.8079±0.2838
5	0.9143±0.3235	0.8422±0.2945	0.9276±0.3208	**0.5265**±0.1611
7	0.9991±0.3505	0.8267±0.2823	1.0100±0.3554	**0.6161**±0.1986
9	1.0317±0.3707	0.8139±0.2827	0.8919±0.3072	**0.6466**±0.2054

is, given a sequence of observation $\mathbf{x} = (x_{i,t})_{i,t} \in \mathbb{R}^{d\times T}$, we draw i.i.d. noise samples $\xi_{i,t} \sim \mathcal{N}(0, \sigma^2)$ and produce noisy input as $\tilde{x}_{i,t} \leftarrow x_{i,t}(1 + \xi_{i,t})$. In data augmentation, we train model with noisy input $\tilde{x}_{i,t}$. In randomized smoothing, the base model is still trained on noisy input $\tilde{x}_{i,t}$ with noise level σ. The noise level σ in the inference phase of randomized smoothing is chosen to be the same as that in data augmentation so there is no need to distinguish the σ used in the two processes.

5.2 Experiment Results

Electricity dataset. Electricity [4] dataset consists of hourly electricity consumption time series from 370 customers. We use the electricity dataset provided by GluonTS [1] and train a DeepVAR model implemented by pytorch-ts [37] with target dimension 10 and rank 5. We choose $\tau = 24$ and $T = 4\tau = 96$, sparsity $k = 1, 3, 5, 7, 9$. In $\mathbf{t} = c_1\hat{x}_{1,T+\tau}$ and $\eta = c_2$, we select $c_1 = 0.5, 2.0$ and $c_2 = 0.5$ respectively and report the largest error produced by these choices of constants. $\sigma = 0.1$ is chosen in data augmentation and randomized smoothing. The metrics under deterministic attack given by Algorithm 1 and probabilistic attack using sparse layer are reported in Table 1 and Table 2 respectively, with attacking configuration $I = \{1\}$ and $H = \{\tau\}$. Besides, we plot wQL under all attacks and defenses against sparsity level to better visualize the effects. See Figure 3a and Figure 3b.

Discussion on Electricity datasets. In this experiment, we have verified the existence of sparse indirect attack, that is, one can attack the prediction of one time series without directly attacking the history of this time series. For example, under deterministic attack, the average wQL is increased by 20% by only attacking one out of nine remaining time series (totally ten but the target time series is excluded). Moreover, attacking half of the time series can increase average wQL by 102%! This observation is even more noticeable under probabilistic attack: average wQL can be increased by 215% with 50% of the time series attacked. In general, average wQL increases as sparsity level increases and probabilistic attack appears to be more effective than deterministic one, see Figure 3a.

As can be seen in Figure 3b, all three defense methods can bring robustness to the forecasting model. Data augmentation and randomized smoothing works well under small sparsity and mini-max defense achieves comparable performance as data augmentation and randomized smoothing under small sparsity and outperforms them under large sparsity.

Additional experiments on Taxi dataset. Taxi dataset is a traffic time series of New York taxi rides taken at 1214 locations for every 30 minutes from January 2015 to January 2016 and considered to be heterogeneous. We use the taxi-30min dataset provided by GluonTS. We train a DeepVAR model with target dimension 10 and rank 5. We choose the same hyper-parameters as in electricity dataset and report the performance of deterministic attack and probabilistic attack in Table 5 and Table 6 respectively in Appendix B. On taxi dataset, it can be observed that in most of the cases under both attacks, our mini-max defense mechanism achieves the best averaged wQL loss.

6 CONCLUSION

In this work, we investigate sparse indirect attack for multivariate time series forecasting models. We propose both deterministic approach and a novel probabilistic approach to finding effective adversarial attack. Besides, we adopt randomized smoothing technique from image classification and univariate time series to our framework and design another mini-max optimization to effectively defend the attack delivered by our attackers. To the best of our knowledge, this is the first work to study sparse indirect attack on multivariate time series and develop corresponding defense mechanisms, which could inspire a future research direction.

REFERENCES

[1] Alexandrov, A., Benidis, K., Bohlke-Schneider, M., Flunkert, V., Gasthaus, J., Januschowski, T., Maddix, D. C., Rangapuram, S. S., Salinas, D., Schulz, J., et al. (2020). Gluonts: Probabilistic and neural time series modeling in python. *J. Mach. Learn. Res.*, 21(116):1–6.

[2] Andersen, T. G., Bollerslev, T., Christoffersen, P., and Diebold, F. X. (2005). Volatility forecasting.

[3] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.

[4] Asuncion, A. and Newman, D. (2007). Uci machine learning repository.

[5] Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

[6] Böse, J.-H., Flunkert, V., Gasthaus, J., Januschowski, T., Lange, D., Salinas, D., Schelter, S., Seeger, M., and Wang, Y. (2017). Probabilistic demand forecasting at scale. *Proceedings of the VLDB Endowment*, 10(12):1694–1705.

[7] Brockwell, P. J. and Davis, R. A. (2009). *Time series: theory and methods*. Springer Science & Business Media.

[8] Brown, R. G. (1957). Exponential smoothing for predicting demand. In *Operations Research*, volume 5, pages 145–145. INST OPERATIONS RESEARCH MANAGEMENT SCIENCES 901 ELKRIDGE LANDING RD, STE ….

[9] Chiang, P.-y., Curry, M., Abdelkader, A., Kumar, A., Dickerson, J., and Goldstein, T. (2020). Detection as regression: Certified object detection with median smoothing. *Advances in Neural Information Processing Systems*, 33:1275–1286.

[10] Cohen, J., Rosenfeld, E., and Kolter, Z. (2019). Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR.

[11] Connor, J., Martin, R., and Atlas, L. (1994). Recurrent neural networks and robust time series prediction. *IEEE Transactions on Neural Networks*, 5(2):240–254.

[12] Croce, F. and Hein, M. (2019). Sparse and imperceivable adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4724–4732.

[13] Dai, H., Li, H., Tian, T., Huang, X., Wang, L., Zhu, J., and Song, L. (2018). Adversarial attack on graph structured data. In *International conference on machine learning*, pages 1115–1124. PMLR.

[14] Dang-Nhu, R., Singh, G., Bielik, P., and Vechev, M. (2020). Adversarial attacks on probabilistic autoregressive forecasting models. In *International Conference on Machine Learning*, pages 2356–2365. PMLR.

[15] de Bézenac, E., Rangapuram, S. S., Benidis, K., Bohlke-Schneider, M., Kurle, R., Stella, L., Hasson, H., Gallinari, P., and Januschowski, T. (2020). Normalizing kalman filters for multivariate time series analysis. *Advances in Neural Information Processing Systems*, 33:2995–3007.

[16] Gasthaus, J., Benidis, K., Wang, Y., Rangapuram, S. S., Salinas, D., Flunkert, V., and Januschowski, T. (2019). Probabilistic forecasting with spline quantile function rnns. In *The 22nd international conference on artificial intelligence and statistics*, pages 1901–1910. PMLR.

[17] Gelper, S., Fried, R., and Croux, C. (2010). Robust forecasting with exponential and Holt–Winters smoothing. *Journal of Forecasting*, 29(3):285–300.

[18] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014a). Generative adversarial nets. *Advances in neural information processing systems*, 27.

[19] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014b). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

[20] Harford, S., Karim, F., and Darabi, H. (2020). Adversarial attacks on multivariate time series. *arXiv preprint arXiv:2004.00410*.

[21] Hu, M. and Root, H. E. (1964). An adaptive data processing system for weather forecasting. *Journal of Applied Meteorology and Climatology*, 3(5):513–523.

[22] Kan, K., Aubet, F.-X., Januschowski, T., Park, Y., Benidis, K., Ruthotto, L., and Gasthaus, J. (2022). Multivariate quantile function forecaster. In *International Conference on Artificial Intelligence and Statistics*, pages 10603–10621. PMLR.

[23] Kim, J., Park, Y., Fox, J. D., Boyd, S. P., and Dally, W. (2020). Optimal operation of a plug-in hybrid vehicle with battery thermal and degradation model. In *2020 American Control Conference (ACC)*, pages 3083–3090. IEEE.

[24] Kumar, A. and Goldstein, T. (2021). Center smoothing: Certified robustness for networks with structured outputs. *Advances in Neural Information Processing Systems*, 34.

[25] Li, B., Chen, C., Wang, W., and Carin, L. (2019). Certified adversarial robustness with additive noise. *Advances in neural information processing systems*, 32.

[26] Lim, B. and Zohren, S. (2021). Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209.

[27] Lim, B., Zohren, S., and Roberts, S. (2020). Recurrent neural filters: Learning independent bayesian filtering steps for time series prediction. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[28] Liu, L. and Zhang, D. (2021a). High-dimensional simultaneous inference on non-gaussian var model via de-biased estimator. *arXiv preprint arXiv:2111.01382*.

[29] Liu, L. and Zhang, D. (2021b). Robust estimation of high-dimensional vector autoregressive models. *arXiv preprint arXiv:2109.10354*.

[30] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

[31] Mode, G. R. and Hoque, K. A. (2020). Adversarial examples in deep learning for multivariate time series regression. In *2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–10. IEEE.

[32] Mudelsee, M. (2019). Trend analysis of climate time series: A review of methods. *Earth-science reviews*, 190:310–322.

[33] Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

[34] Park, Y., Maddix, D., Aubet, F.-X., Kan, K., Gasthaus, J., and Wang, Y. (2022). Learning quantile functions without quantile crossing for distribution-free time series forecasting. In *International Conference on Artificial Intelligence and Statistics*, pages 8127–8150. PMLR.

[35] Park, Y., Mahadik, K., Rossi, R. A., Wu, G., and Zhao, H. (2019). Linear quadratic regulator for resource-efficient cloud services. In *Proceedings of the ACM Symposium on Cloud Computing*, pages 488–489.

[36] Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., and Januschowski, T. (2018). Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31.

[37] Rasul, K. (2021). PytorchTS.

[38] Salinas, D., Bohlke-Schneider, M., Callot, L., Medico, R., and Gasthaus, J. (2019). High-dimensional multivariate forecasting with low-rank gaussian copula processes. *Advances in neural information processing systems*, 32.

[39] Salinas, D., Flunkert, V., Gasthaus, J., and Januschowski, T. (2020). Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191.

[40] Salman, H., Li, J., Razenshteyn, I., Zhang, P., Zhang, H., Bubeck, S., and Yang, G. (2019). Provably robust deep learning via adversarially trained smoothed classifiers. *Advances in Neural Information Processing Systems*, 32.

[41] Sen, R., Yu, H.-F., and Dhillon, I. S. (2019). Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. *Advances in neural information processing systems*, 32.

[42] Singh, G., Gehr, T., Mirman, M., Püschel, M., and Vechev, M. (2018). Fast and effective robustness certification. *Advances in neural information processing systems*, 31.

[43] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

[44] Taxi, N. and Commission, L. (2015). Tlc trip record data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[45] Taylor, S. J. and Letham, B. (2018). Forecasting at scale. *The American Statistician*, 72(1):37–45.

[46] Wang, D. and Tsay, R. S. (2021). Robust estimation of high-dimensional vector autoregressive models. *arXiv preprint arXiv:2107.11002*.

[47] Wang, Y., Smola, A., Maddix, D., Gasthaus, J., Foster, D., and Januschowski, T. (2019). Deep factors for forecasting. In *International conference on machine learning*, pages 6607–6617. PMLR.

[48] Wen, Q., Sun, L., Yang, F., Song, X., Gao, J., Wang, X., and Xu, H. (2020). Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478*.

[49] Weng, L., Chen, P.-Y., Nguyen, L., Squillante, M., Boopathy, A., Oseledets, I., and Daniel, L. (2019). Proven: Verifying robustness of neural networks with a probabilistic approach. In *International Conference on Machine Learning*, pages 6727–6736. PMLR.

[50] Weng, L., Zhang, H., Chen, H., Song, Z., Hsieh, C.-J., Daniel, L., Boning, D., and Dhillon, I. (2018). Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pages 5276–5285. PMLR.

[51] Wong, E. and Kolter, Z. (2018). Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR.

[52] Wong, E., Rice, L., and Kolter, J. Z. (2020). Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*.

[53] Yoon, T., Park, Y., Ryu, E. K., and Wang, Y. (2022). Robust probabilistic time series forecasting. *arXiv preprint arXiv:2202.11910*.

[54] Zhai, R., Dan, C., He, D., Zhang, H., Gong, B., Ravikumar, P., Hsieh, C.-J., and Wang, L. (2020). Macer: Attack-free and scalable robust training via maximizing certified radius. *arXiv preprint arXiv:2001.02378*.

[55] Zhang, J., Lou, Y., Wang, J., Wu, K., Lu, K., and Jia, X. (2021). Evaluating adversarial attacks on driving safety in vision-based autonomous vehicles. *arXiv preprint arXiv:2108.02940*.

A MORE METRICS ON ELECTRICITY DATASET

To measure the performance of a forecasting model, other metrics like Weighted Absolute Percentage Error (WAPE) or Weighted Squared Error (WSE) are also considered by a large body of literature. For completeness, we present the definition of WAPE and WSE:

$$\text{WAPE} = \sum \left| \frac{\text{predicted value}}{\text{true value}} - 1 \right| = \frac{1}{|I||H|} \sum_{i \in I, h \in H} \left| \frac{\frac{1}{n} \sum_{j=1}^{n} \hat{x}^j_{T+h,i}}{x_{T+h,i}} - 1 \right|$$

$$\text{WSE} = \sum \left(\frac{\text{predicted value}}{\text{true value}} - 1 \right)^2 = \frac{1}{|I||H|} \sum_{i \in I, h \in H} \left(\frac{\frac{1}{n} \sum_{j=1}^{n} \hat{x}^j_{T+h,i}}{x_{T+h,i}} - 1 \right)^2$$

We report WAPE, WSE and wQL under deterministic and probabilistic attacks on electricity dataset in Table 3 and Table 4.

Table 3: Metrics on electricity dataset under deterministic attack. Target time series $I = \{1\}$ and attacked time stamp $H = \{\tau\}$. Smaller is better.

	no defense			data augmentation			randomized smoothing			mini-max defense		
Sparsity	WAPE	WSE	wQL	WAPE	WSE	wQL	WAPE	WSE	wQL	WAPE	WSE	wQL
no attack	0.3784±0.2435	0.2025±0.2633	0.2853±0.0825	0.2975±0.1421	0.1087±0.0727	0.2288±0.0792	0.2810±0.1445	0.0998±0.0675	0.2176±0.0700	0.2913±0.1362	0.1034±0.0724	**0.2154**±0.0705
1	0.4707±0.2849	0.3028±0.3628	0.3410±0.0946	0.4128±0.2658	0.2411±0.2619	0.2949±0.0716	0.3936±0.2442	0.2146±0.2145	**0.2826**±0.0718	0.4231±0.2598	0.2465±0.2150	0.2990±0.0772
3	0.6637±0.4436	0.6372±0.7861	0.4559±0.1344	0.5533±0.2452	0.3663±0.2654	**0.3655**±0.1097	0.5564±0.2616	0.3780±0.2895	0.3757±0.1012	0.5847±0.3796	0.4859±0.4365	0.3775±0.0923
5	0.8167±0.5127	0.9299±1.0085	0.5770±0.1772	0.7926±0.5421	0.9221±1.0479	0.5554±0.1636	0.7924±0.5366	0.9158±1.0489	0.5560±0.1751	0.8003±0.5162	0.9070±0.8311	**0.5273**±0.1558
7	0.9353±0.6183	1.2572±1.3999	0.6687±0.2131	0.9947±0.6051	1.3556±1.3441	0.7076±0.2321	0.9945±0.6367	1.3944±1.4845	0.7072±0.2308	0.9299±0.5439	1.1606±0.9590	**0.6506**±0.2111
9	1.1140±0.7385	1.7865±1.9223	0.8282±0.2847	1.1474±0.7336	1.8546±1.9234	0.8412±0.2896	1.1454±0.7297	1.8444±1.9262	0.8327±0.2786	1.0502±0.6403	1.5128±1.2721	**0.7503**±0.2588
full attack	1.3696±0.9286	2.7383±3.1382	1.0159±0.3542	1.0765±0.6046	1.5243±1.2691	**0.8023**±0.2664	1.1048±0.6350	1.6238±1.4510	0.8133±0.2659	1.2567±0.7118	2.0860±1.6019	0.9415±0.3424

Table 4: Metrics on electricity dataset under probabilistic attack using sparse layer. Target time series $I = \{1\}$ and attacked time stamp $H = \{\tau\}$. Smaller is better.

	No defense			data augmentation			randomized smoothing			mini-max defense		
Sparsity	WAPE	WSE	wQL	WAPE	WSE	wQL	WAPE	WSE	wQL	WAPE	WSE	wQL
no attack	0.3842±0.2620	0.2162±0.3044	0.2909±0.0748	0.3074±0.1746	0.1250±0.0946	0.2374±0.0764	0.2858±0.1547	0.1056±0.0761	**0.2237**±0.0750	0.3218±0.1429	0.1240±0.0830	0.2342±0.0710
1	0.6230±0.6324	0.7881±1.1864	**0.4364**±0.1296	0.7476±0.7240	1.0830±1.8593	0.5923±0.0913	0.7683±0.8771	1.3596±2.7290	0.5940±0.1142	0.6990±0.6957	0.9726±1.7182	0.4935±0.1450
3	1.0540±0.7522	1.6768±1.4810	0.7245±0.2434	0.8484±0.6809	1.1834±1.3998	0.5738±0.1759	0.6784±0.5230	0.7337±0.7698	**0.4581**±0.1301	0.9909±0.7564	1.5540±1.8925	0.8079±0.2838
5	1.2078±0.7451	2.0139±2.0667	0.9143±0.3235	1.1444±0.6665	1.7538±1.4318	0.8422±0.2945	1.2310±0.7025	2.0090±1.6609	0.9276±0.3208	0.6966±0.4554	0.6927±0.8752	**0.5265**±0.1611
7	1.3236±0.7310	2.2863±1.8336	0.9991±0.3505	1.1304±0.6522	1.7031±1.4053	0.8267±0.2823	1.3496±0.6777	2.2809±1.7240	1.0100±0.3554	0.8424±0.7803	1.3186±1.7286	**0.6161**±0.1986
9	1.3656±0.8671	2.6166±2.6679	1.0317±0.3707	1.0912±0.6181	1.5727±1.2081	0.8139±0.2827	1.1978±0.6742	1.8894±1.5309	0.8919±0.3072	0.8691±0.7410	1.3043±2.0663	**0.6466**±0.2054

B ADDITIONAL EXPERIMENTS ON TAXI DATASET

Table 5: Metrics on taxi dataset under deterministic attack. Target time series $I = \{1\}$ and attacked time stamp $H = \{\tau\}$. Smaller is better.

	no defense			data augmentation			randomized smoothing			mini-max defense		
Sparsity	WAPE	WSE	wQL	WAPE	WSE	wQL	WAPE	WSE	wQL	WAPE	WSE	wQL
no attack	3.2142±2.4891	16.5265±26.0154	0.6320±0.1114	3.4168±2.4399	17.6273±27.2472	0.6683±0.1447	3.4254±2.4796	17.8821±27.2862	0.6737±0.1351	2.9806±2.4607	14.9394±26.4364	**0.5740**±0.1256
1	3.2829±2.5329	17.1934±25.8223	0.6402±0.1087	3.6419±2.5344	19.6865±27.7478	0.7110±0.1299	3.5855±2.4891	19.0519±26.6860	0.7001±0.1268	2.9807±2.3575	14.4425±23.0432	**0.5675**±0.1218
3	3.8502±2.7065	22.1489±28.6597	0.7431±0.1298	4.1094±2.6854	24.0987±30.2943	0.7927±0.1281	4.0566±2.6615	23.5399±28.7010	0.7867±0.1328	3.3575±2.3293	16.6983±22.6643	**0.6187**±0.1357
5	4.6638±2.9978	30.7378±34.5478	0.8994±0.1836	4.8021±2.9732	31.8996±35.2534	0.9311±0.1728	4.7566±2.9546	31.3546±34.3349	0.9239±0.1653	3.8960±2.7201	22.5774±30.0041	**0.7224**±0.1583
7	5.2899±3.4284	39.7373±44.1055	1.0409±0.2378	5.4514±3.3364	40.8489±41.8463	1.0580±0.2157	5.4342±3.4091	41.1526±43.6852	1.0602±0.2106	4.5777±3.0506	30.2612±35.2095	**0.8601**±0.2028
9	5.7683±3.7389	47.2523±50.1275	1.1368±0.2766	5.9307±3.8472	49.9744±55.4119	1.1560±0.2544	5.9214±3.6963	48.7254±50.5991	1.1489±0.2535	5.0326±3.3762	36.7260±40.8499	**0.9552**±0.2342
full attack	5.7031±3.6022	45.5005±45.5102	1.1229±0.2677	5.7006±3.4781	44.5936±44.9340	1.1004±0.2438	5.7582±3.5435	45.7137±46.1601	1.1256±0.2500	4.4553±3.2047	30.1198±35.0223	**0.8572**±0.1947

Table 6: Metrics on taxi dataset under probabilistic attack. Target time series $I = \{1\}$ and attacked time stamp $H = \{\tau\}$. Smaller is better.

	no defense			data augmentation			randomized smoothing			mini-max defense		
Sparsity	WAPE	WSE	wQL	WAPE	WSE	wQL	WAPE	WSE	wQL	WAPE	WSE	wQL
no attack	3.2024±2.5543	16.7798±27.0240	0.6331±0.1147	3.4415±2.4311	17.7543±26.9227	0.6739±0.1416	3.3960±2.4740	17.6534±27.1673	0.6658±0.1407	2.9347±2.4405	14.5687±25.5719	**0.5664**±0.1180
1	3.5647±2.6634	19.8008±26.9725	0.6974±0.1150	3.5624±2.4657	18.7701±25.6556	0.6957±0.1258	3.5970±2.4377	18.8808±25.4870	0.7009±0.1292	3.0822±2.4489	15.4971±25.1302	**0.5926**±0.1230
3	3.5497±2.8747	20.8643±31.7045	0.6947±0.1163	3.7124±2.7975	21.6080±29.1594	0.7250±0.1201	3.8457±2.6817	21.9809±26.7964	0.7460±0.1275	3.2417±2.6009	17.2736±23.5462	**0.6414**±0.1180
5	4.2499±2.8963	26.4501±30.1929	0.8080±0.1527	3.9792±2.6460	22.8357±26.4122	**0.7752**±0.1345	4.1840±2.7878	25.2774±28.4534	0.7935±0.1478	3.1440±2.5098	16.1838±21.2503	0.7900±0.1407
7	4.7907±3.0940	32.5235±35.2534	0.9334±0.2086	3.8134±2.9295	23.1238±34.0380	0.7314±0.1385	4.2922±2.8123	26.3319±36.2382	0.8044±0.1647	3.2961±2.6990	18.1486±27.0921	**0.6958**±0.1255
9	5.3450±3.3176	39.5753±38.6771	1.0470±0.2455	4.9965±2.9362	33.5863±31.8005	0.9637±0.1977	5.3693±3.0930	38.3966±35.8923	1.0286±0.2111	3.2063±2.6314	17.2046±25.2366	**0.7292**±0.1229