MIA: A Framework for Certifiably Robust Time-Series Classification and Forecasting Against Temporally-Localized Perturbations

Anonymous authors
Paper under double-blind review

Abstract

Recent literature demonstrates that times-series forecasting/classification are sensitive to input perturbations. However, the defenses for time-series models are relatively under-explored. In this paper, we propose **M**asking **I**mputing **A**ggregation (MIA), a plug-and-play framework to provide an arbitrary deterministic time-series model with certified robustness against temporally-localized perturbations (also known as ℓ_0-norm localized perturbations), which is to our knowledge the first ℓ_0-norm defense for time-series models. Our main insight is to let an occluding mask move across the input series, guaranteeing that, for an arbitrary localized perturbation there must exist at least a mask that completely remove out the perturbation, so that our prediction on this masked series is uninfluenced. Remarkably, MIA is high-availability as it still works even if we only have query access to the pretrained model. Moreover, as there is no dedicated defense against ℓ_0-norm perturbations for time-series models, we specifically adapt two matrix-based defenses to time-series models for comparison. Extensive experiments show that MIA yields stronger robustness as well as practicality.

1 Introduction

Time series forecasting/classification (TSF/TSC) have been widely applied to help businesses make informed decisions and plans (Miyato et al., 2017; Zhou et al., 2019; Schlegl et al., 2019; Park et al., 2018). However, a wide range of literature demonstrate that time-series models are vulnerable to adversarial input perturbations (Connor et al., 1994; Gelper et al., 2010; Ding et al., 2022; Yang et al., 2020; Dang-Nhu et al., 2020; Oregi et al., 2018; Han et al., 2020), e.g., an elaborately designed imperceptible perturbation could control the prediction. So far related literature is mainly focusing on detecting the outliers (Ruff et al., 2018; Yairi et al., 2017), the adversarial robustness of time-series models is relatively under-explored, especially ℓ_0-norm robustness, e.g., (Yoon et al., 2022) only explore the ℓ_2-norm adversarial robustness for probabilistic forecasting models. In the present work, we investigate the ell_0-robustness against localized perturbations, as we notice there already exists corresponding powerful attacks (Yang et al., 2022).

In terms of heuristic defense approaches, they can yield better empirical robustness but lack robustness guarantees. According to the experience from the heuristic defenses for the computer vision tasks (Athalye et al., 2018; Carlini & Wagner, 2017; Athalye & Carlini, 2018) (e.g., Athalye et al. (Athalye et al., 2018) leverage BPDA technique to circumvent most prior heuristic defenses successfully), we have reason to worry that these heuristic defenses would be invalidated when confronted with the newly-designed adaptive attacks. To end such a "cat and mouse" game between the adaptive attacks and the heuristic defenses, the concept of certified defense is proposed. guaranteeing the robustness for each prediction mathematically.

In terms of certified defenses, they can produce robustness certificates but often require the user to train the base model from scratch, e.g., (Yoon et al., 2022; Li et al., 2020; Cohen et al., 2019) need to train the model on Gaussian augmented data for an acceptable performance, since these defenses are often ineffective on the natural models. Such requirement may bring additional uncertainties and challenges in real-world deployments. Moreover, so far almost all the certified defenses are focusing on matrix-based data (e.g., image), the defenses for sequence-based data are quite terra-incognita.

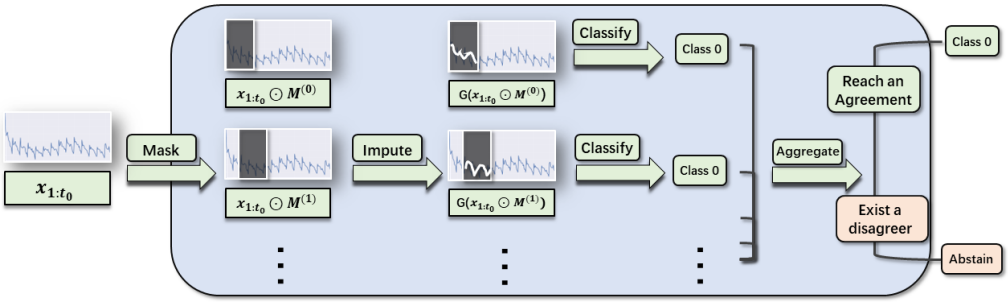


Figure 1: Overview of MIA pipeline. Inputted a series $\mathbf{x}_{1:t_0}$, MIA first masks different periods of $\mathbf{x}_{1:t_0}$ to construct the masked series $\mathbf{x}_{1:t_0} \odot \mathrm{M}^{(m)}, m = 0, \ldots, M$. Then MIA imputes the masked series with the imputation model $G(\cdot)$. We classify the imputed series with the pretrained model. If the predictions of all the imputed series are *Class 0*, MIA will return *Class 0* with the robustness guarantee that the output is clean, otherwise MIA will return **Abstain**.

To address these issues, in this paper, we propose **M**asking **I**mputing **A**ggregation (MIA), a plug-and-play framework for all the TSF/TSC deterministic models to provide robustness guarantees against temporally-localized perturbations. MIA only requires an imputation model additionally for imputing the masked series, which can be trained in an unsupervised manner. Specifically, MIA works as follows: 1) **masking:** MIA first generate a list of masked series via moving a mask across the input series, with the guarantee that there is no temporally-localized perturbation (w.r.t. ℓ_0-norm constraint) can influence all the masked series simultaneously; 2) **imputing:** MIA then imputes the masked series with an imputation model; 3) **aggregation:** MIA will return the result only if the pretrained model outputs the same prediction on all the imputed series. Finally, we can guarantee that the final output must equal the prediction on the uninfluenced masked series. Furthermore, we compare MIA to derandomized smoothing (Levine & Feizi, 2020a) and randomized ablation (Levine & Feizi, 2020b), two randomized smoothing based defenses that are firstly adapted to time-series models by us. **The contributions are: 1)** We propose MIA, a plug-and-play framework to produce robustness certificates against temporally-localized perturbations for both TSF and TSC deterministic models, which is to our knowledge the first ℓ_0-norm certified defense for time-series models.

2) We propose masked training, a training algorithm specialized for the imputation models of MIA, which exploits the nature of consecutive masked series. Experiments validate that the imputation models trained by masked training can yield higher robustness.

3) We firstly adapt two matrix-based certified defenses (used in vision) to the time-series defenses. In comparison, MIA is more practical because MIA can be directly applied to the pretrained models, while the other two defenses need to train the base model over noisy samples in a supervised setting. Extensive experiments validate that MIA can offer stronger robustness.

2 Related Work

Heuristic defenses for time-series models. Prior works on robust TSF/TSC can be divided into two general categories: traditional outlier-based approaches and deep learning approaches. The former is to filter the outliers in a statistical way, including k-Means clustering (Yang et al., 2017), one-class SVM clustering (Schölkopf et al., 2001), Kalman filters (de Bézenac et al., 2020) and support vector data description (Tax & Duin, 2004). The latter leverages the strong representation ability of neural networks, including robust feature-based approaches (Guo et al., 2016; Yang & Fan, 2022), reconstruction-based methods (Li et al., 2021; 2019; Xu et al., 2018; Schlegl et al., 2019), GNN-based methods (Zhao et al., 2020; Deng & Hooi, 2021), association discrepancy (Xu et al., 2022), LSTM-based methods (Hundman et al., 2018; Tariq et al., 2019). However, all of these methods lack robustness guarantees, hinting that they could become invalid once the adversary designs a new adaptive attack to bypass them. For that reason, certified defenses are crucial since their mathematical robustness certificates are permanently valid.

Certified adversarial defenses. In the field of image classification, there has been much work on the certified adversarial defenses, including randomized smoothing (Cohen et al., 2019; Salman et al.,

2020), convex polytope (Wong & Kolter, 2018), CROWN-IBP (Zhang et al., 2019) and Lipschitz bounding (Cisse et al., 2017). In particular, the certified defenses of the similar objective to our work (ℓ_0-norm robustness) include derandomized smoothing (Levine & Feizi, 2020a), randomized ablation (Levine & Feizi, 2020b; Zhang et al., 2020) and a series of mask-based defenses (Xiang & Mittal, 2021; McCoyd et al., 2020; Han et al., 2021; Xiang et al., 2021; 2022). In stark contrast, the certified defenses for time-series models are rarely explored. To our knowledge, (Yoon et al., 2022) and (Li et al., 2020) are the only two defenses that produce ℓ_2-norm robustness certificates, but a common downside is that they require the base model to perform well over Gaussian augmented samples, as most natural models do not meet this requirement. Therefore, we need to train the base model on noisy samples from scratch, imposing a large amount of additional training costs.

3 Preliminaries and Problem Setup

Time series classification (TSC). First we model the task of time series classification. Specifically, inputted a time series of length t_0 (denoted by $\mathbf{x}_{1:t_0} = [x_1, x_2, \ldots, x_{t_0}]$), TSC model returns the class of the inputted series (denoted by $f(\cdot) : \mathbf{x}_{1:t_0} \to y$).

Time series forecasting (TSF). Given "past" (denoted by $\mathbf{x}_{1:t_0}$), the forecasting model returns the "future" (denoted by $\mathbf{x}_{t_0+1:t_0+\tau}$). For ease of understanding, we consider the classic and commonly studied short-term forecasting setting (Ke et al., 2017): forecasting a single time point (not necessarily the next time point x_{t_0+1}) $f(\cdot) : \mathbf{x}_{1:t_0} \to \mathbb{R}$. Since the problem of long-term forecasting $f(\mathbf{x}_{1:t_0}) \to \mathbf{x}_{t_0+1:t_0+\tau}$ can be decomposed into τ sub-problems of short-term forecasting, of which the i-th ($i = 1, \ldots, \tau$) forecaster predicts the time point x_{t_0+i}.

Definition 1 (Temporally-localized perturbation)**.** *Temporally-localized perturbation is to* perturb consecutive time points among the input series w.r.t. ℓ_0-norm constraint*. The perturbed series is:*

$$\begin{aligned}
&\boldsymbol{x}_{1:t_0} + \delta, \quad \textit{subject to} \quad \|\delta\|_0 \leq L_{\text{adv}} \\
=&\boldsymbol{x}_{1:t_0} + [0, \ldots, 0, \delta_{t_{\text{adv}}+1}, \ldots, \delta_{t_{\text{adv}}+L_{\text{adv}}}, 0, \ldots, 0] \\
=&[x_1, \ldots, \underbrace{x_{t_{\text{adv}}+1} + \delta_{t_{\text{adv}}+L_{\text{adv}}}, \ \ldots, \ x_{t_{\text{adv}}+L_{\text{adv}}} + \delta_{t_{\text{adv}}+1}}_{\textit{Perturbed subsequence}}, \ \ldots, x_{t_0}]
\end{aligned} \tag{1}$$

where $t_{\text{adv}} + 1$ and L_{adv} denote the starting point and the ℓ_0 norm of the perturbation.

Temporally-localized perturbation corresponds to the attack scenario where the time-series data is influenced by a short-term event, which is representative in the real world, e.g., the stock price drops abnormally for a short period due to the CEO scandal. We believe that, in most cases, a short-term event should not have a large impact on the long-term outcomes, thus enhancing the resistance of time-series models to temporally-localized perturbations is essential. To highlight the risk of temporally-localized perturbations, we propose an efficient generation algorithm and show how much a perturbation can change the forecasting results of undefended forecasters in Appendix.

4 Proposed Framework: Masking Imputing Aggregation

4.1 Pipeline Overview

MIA includes three steps: 1) masking; 2) imputing; 3) aggregation.

1. **Masking.** Inputted a testing time series $\mathbf{x}_{1:t_0}$, we generate the masked series $\hat{\mathbf{x}}_{1:t_0} = \mathbf{x}_{1:t_0} \odot \mathrm{M}(t_{\text{mask}}, L_{\text{mask}})$, where $\odot\mathrm{M}(t_{\text{mask}}, L_{\text{mask}})$ is to remove out the values of the time period $[t_{\text{mask}}+1 : \min(t_{\text{mask}} + L_{\text{mask}}, t_0)]$. Specifically, fixing L_{adv} and L_{mask}, the masked series (denoted by $\hat{\mathbf{x}}^{(m)}_{1:t_0}, m = 0, \ldots, M$) are constructed as follows: [1]

$$\begin{aligned}
&\hat{\mathbf{x}}^{(m)}_{1:t_0} = \mathbf{x}_{1:t_0} \odot \mathrm{M}(1 + m\alpha, L_{\text{mask}}), \ m = 0, \ldots, M \\
&\text{where} \quad \alpha = L_{\text{mask}} - L_{\text{adv}} + 1, \ M = \lfloor (t_0 - 1)/\alpha \rfloor
\end{aligned} \tag{2}$$

where α is the step size of our mask moving. The number of the masked series is $M + 1$.

2. **Imputing.** Denote $G(\cdot)$ a time-series imputation model (we will discuss $G(\cdot)$ later this section). Our second step is to recover the masked values of $\hat{\mathbf{x}}^{(m)}_{1:t_0}$ with $G(\cdot)$, as follows:

$$\mathbf{x}^{(m)}_{1:t_0} = G(\mathbf{x}^{(m)}_{1:t_0}) \tag{3}$$

[1] $\lfloor c \rfloor$ returns the largest integer smaller than or equal to c

Algorithm 1: Algorithm of Masking Imputing Aggregation.

Input: The pretrained TSF/TSC model $f(\cdot)$, the imputation model $G(\cdot)$, the input series $\mathbf{x}_{1:t_0}$, the mask size L_{mask}, the length of temporally-localized perturbation L_{adv}, the discretization parameter Δ for TSF task.

1 Compute the step size of masking $\alpha \leftarrow L_{\text{mask}} - L_{\text{adv}} + 1$;

2 Generate the masked series $\hat{\mathbf{x}}_{1:t_0}^{(m)} \leftarrow \mathbf{x}_{1:t_0} \odot \mathcal{M}^{(m)}, m = 0, 1, \ldots, M$;

$\mathbf{x}_{1:t_0} \odot \mathcal{M}^{(m)}$ is computed as Eq. (2) ;

3 Utilize the imputation model to impute the masked series $\mathbf{x}_{1:t_0}^{(m)} \leftarrow g(\hat{\mathbf{x}}_{1:t_0}^{(m)})$;

4 Compute the output (denoted by $y^{(m)}$) for each imputed series, as follows:

$$y^{(m)} = \begin{cases} f(\mathbf{x}_{1:t_0}^{(m)}) & f(\cdot) \text{ is a TSC model} \\ f_{\text{dis}}(\mathbf{x}_{1:t_0}^{(1)}) & f(\cdot) \text{ is a TSF model} \end{cases}$$

Here $f_{\text{dis}}(\mathbf{x}_{1:t_0}^{(m)})$ is computed as Eq. (5);

5 **if** $y^{(0)} = y^{(1)} = \ldots = y^{(M)}$ **then**

  Output: $y^{(0)}$.

6 **else**

  Output: Abstain.

This step is to make $\mathbf{x}_{1:t_0}^{(m)}$ approximate the normal time series, so that the pretrained model perform similarly on these imputed series.

3. **Aggregation.** We input $\mathbf{x}_{1:t_0}^{(m)}$ into the pretrained model $f(\cdot)$. If the pretrained model's ouputs on all $\mathbf{x}_{1:t_0}^{(m)}$ reach agreement unanimously, MIA (denoted by $f_{\text{MIA}}(\mathbf{x}_{1:t_0})$) will output this unanimously approved label/prediction, otherwise will output **Abstain** to alert that the input might have been attacked by the temporally-localized perturbations.

$$f_{\text{MIA}}(\mathbf{x}_{1:t_0}) = \begin{cases} f(\mathbf{x}_{1:t_0}^{(0)}) & f(\mathbf{x}_{1:t_0}^{(0)}) = f(\mathbf{x}_{1:t_0}^{(1)}) = \ldots = f(\mathbf{x}_{1:t_0}^{(M)}) \\ \textbf{Abstain} & \text{Otherwise} \end{cases} \tag{4}$$

Discretization technique. We notice that, it is almost impossible for TSF models to forecast exactly the same value on $M + 1$ different imputed series, causing MIA to output **Abstain** all the time. To address this problem, we add a layer after the forecaster to discretize its outputs. Given the pretrained model $f(\dot{c})$, the discretized forecaster (denoted by $f_{\text{dis}}(\cdot)$) works as follows:

$$f_{\text{dis}}(\mathbf{x}_{1:t_0}) = \Delta \lfloor f(\mathbf{x}_{1:t_0}) / \Delta \rfloor \tag{5}$$

where Δ is a discretization parameter to control the trade-off between the retained information after discretization and the success rate of achieving the agreement. As Δ increases, the discretized forecasts retain more information from the original forecasts while the agreement rate decreases.

Remark 1 (**Mask Size** L_{mask})**.** *The only necessary requirement of masking (step 1) is to ensure that* for an arbitrary temporally-localized perturbation of L_{adv}, there exists at least one mask that occludes the perturbation completely*, implying that* L_{mask} *must be no smaller than* L_{adv}*. As* L_{mask} *increases, the imputation quality decreases as we need to impute more missing values. Meanwhile, the number of masked series linearly decreases with the mask size, suggesting that large* L_{mask} *helps save inference cost. In the extreme case where* $L_{\text{mask}} = t_0$*, we have* $\hat{\boldsymbol{x}}^{(0)} = \hat{\boldsymbol{x}}^{(1)} = \ldots = \hat{\boldsymbol{x}}^{(M)} = \mathbf{0}_{1:t_0}$*. MIA always outputs* $f(\mathbf{0}_{1:t_0})$ *regardless of the input series, thus the inference cost is* 0*.*

Remark 2 (MIA on Probabilistic Models)**.** *A line of time-series forecasting models are the probabilistic models (e.g., DeepAR (Salinas et al., 2020)), which forecast over a random distribution (denoted by* $q[y \mid \boldsymbol{x}_{1:t_0}]$*), as follows:*

$$f(\boldsymbol{x}_{1:t_0}) = \mathbb{E}_{q[x_{t_0+1} | \boldsymbol{x}_{1:t_0}]} \left[x_{t_0+1} \right] \tag{6}$$

Applying MIA to probabilistic models is challenging because the exact value of forecast is inaccessible, hindering our aggregation (Step 3). Although we can utilize Clopper-Pearson estimation

method (Clopper & Pearson, 1934) to estimate $f_{\text{dis}}(\boldsymbol{x}_{1:t_0})$ with a confidence level, the inference cost on probabilistic models is much higher than deterministic models for the confidence interval estimation, reducing the practicality of MIA. [2]

4.2 Implementation of MIA

The practical implementation of MIA is showed in Algorithm 1.

4.3 Robustness Certificate of MIA

Proposition 1 (Robustness Certificate of MIA)**.** *The forecast/label (not **Abstain**) returned by Algorithm 1 cannot be changed by any temporally-localized perturbations whose ℓ_0 norm is no larger than L_{adv} (see proof in Appendix).*

Remark 3 (**Robustness Certificate**)**.** *We emphasize that, our certificate is only for $f_{\text{MIA}}(\boldsymbol{x}_{1:t_0})$, instead of $f(\boldsymbol{x}_{1:t_0})$, as $f_{\text{MIA}}(\boldsymbol{x}_{1:t_0})$ and $f(\boldsymbol{x}_{1:t_0})$ are not necessarily equal. To certify the robustness for a sequence of time series $\boldsymbol{x}_{t_0+1:t_0+\tau}$, we just need to implement the algorithm 1 on $x_{t_0+1}, x_{t_0+2}, \ldots, x_{t_0+\tau}$ one by one, to certify the robustness for each time point.*

4.4 Training Imputation Model $G(\cdot)$

The performance of MIA frameworks highly depends on the imputation model $G(\cdot)$ that we use. We notice that there exists much work on time series imputation (Cao et al., 2018; Du et al., 2022; Moritz & Bartz-Beielstein, 2017; Fortuin et al., 2020; Cao et al., 2018; Luo et al., 2019; Yozgatligil et al., 2013). However, all these imputation models aim to recover the discrete missing values, thus perform poorly on imputing a consecutive sequence of missing values. To address this problem, we propose *randomized masked training* to train the imputation model $G(\cdot)$, which is to minimize the MSE loss over $M+1$ masked series with Gaussian augmentation, as follows:

$$\mathbb{E}_{\delta_{1:t_0}\sim\mathcal{N}(0,\sigma^2)} \ \frac{1}{M+1}\sum_{m=0}^{M} \|G\left((\mathbf{x}_{1:t_0}+\delta_{1:t_0})\odot \mathbf{M}^{(m)}\right)-\mathbf{x}_{1:t_0}\|_2^2 \tag{7}$$

where $\delta_{1:t_0}\sim\mathcal{N}(0,\sigma^2)$ is a length-t_0 vector, of which each entry is *i.i.d.* from Gaussian $\mathcal{N}(0,\sigma^2)$. We adopt MLP-Mixer (Tolstikhin et al., 2021) as the architecture of our imputation model.

4.5 Comparison to Randomized Smoothing Defenses

We show how to extend derandomized smoothing (DS) (Levine & Feizi, 2020a) and randomized ablation (RA) (Levine & Feizi, 2020b), to the certified defenses against temporally-localized perturbations. We compare MIA to these two defenses.

Derandomized smoothing for time-series models. In the time-series version of DS, given a time series $\mathbf{x}_{1:t_0}$ and the base classifier $f(\cdot)$, DS (denoted by f_{DS}) classifies as follows[3]:

$$f_{\text{DS}}(\mathbf{x}_{1:t_0}) = \underset{y\in\mathcal{Y}}{\arg\max}\left[\sum_{\mathbf{x}_{\text{sub}}\in\text{Sub}(\mathbf{x}_{1:t_0},k)} \mathbb{I}\{f(\mathbf{x}_{\text{sub}})=y\}\right] \tag{8}$$

where $\text{Sub}(\mathbf{x}_{1:t_0},k)$ consists of the subsequences $\mathbf{x}_{1:k}, \mathbf{x}_{k+1:2k}, \mathbf{x}_{2k+1:3k}, \ldots, \mathbf{x}_{t_0-k+1,t_0}$. We first let the base classifier make predictions on these subsequences, and then $f_{\text{DS}}(\mathbf{x}_{1:t_0})$ outputs the majority label. The prediction $\hat{y}=f_{\text{DS}}(\mathbf{x}_{1:t_0})$ is provably robust to temporally-localized perturbations (ℓ_0 norm is L_{adv}) if

$$\sum_{\mathbf{x}_{\text{sub}}\in\text{Sub}(\mathbf{x}_{1:t_0},k)} \mathbb{I}\{f(\mathbf{x}_{\text{sub}})=\hat{y}\} - \max_{y\neq\hat{y}} \sum_{\mathbf{x}_{\text{sub}}\in\text{Sub}(\mathbf{x}_{1:t_0},k)} \mathbb{I}\{f(\mathbf{x}_{\text{sub}})=y\} > k+L_{\text{adv}}-1 \tag{9}$$

[2] We notice that a recent work (Yoon et al., 2022) derives robustness certificate for probabilistic forecasters, but our definitions of robustness are different. (Yoon et al., 2022) is to bound the sensitivity to the perturbation, while our robustness objective is stricter, aiming to guarantee that the forecast is invariant to the perturbation.

[3] $\mathbb{I}\{\}$ is the indicator function.

Table 1: (DistalPhalanxTW) Comparison among three defenses on a TSC dataset.

Defense	1 %	2 %	3 %	4 %	5 %	6 %	7 %	8 %	9 %	10 %
MIA ($L_{\text{mask}} = 10\%$)	**67.3%**	**66.3%**	**67.3%**	**67.3%**	**66.3%**	**64.4%**	**66.3%**	**64.4%**	**64.4%**	**63.4%**
DS ($k = 10\%$)	28.1%	28.1%	28.1%	28.1%	28.1%	28.1%	28.1%	28.1%	28.1%	28.1%
RA ($k = 10\%$)	5.8%	5.8%	5.8%	5.8%	5.8%	5.8%	5.8%	5.8%	5.8%	5.8%
MIA ($L_{\text{mask}} = 15\%$)	**62.4%**	**65.3%**	**65.3%**	**64.4%**	**64.4%**	**62.4%**	**63.4%**	**65.3%**	**63.4%**	**64.4%**
DS ($k = 15\%$)	30.2%	30.2%	30.2%	30.2%	30.2%	30.2%	30.2%	30.2%	30.2%	30.2%
RA ($k = 15\%$)	9.4%	9.4%	9.4%	9.4%	9.4%	9.4%	9.4%	9.4%	9.4%	9.4%

Randomized ablation for time-series models. RA (denoted by $f_{\text{RA}}(\cdot)$) classifies as follows:

$$f_{\text{RA}}(\mathbf{x}_{1:t_0}) = \arg\max_{y \in \mathcal{Y}} \left[\Pr_{\mathbf{x}_{\text{sub}} \sim \text{Sample}(\mathbf{x}_{1:t_0}, k)} [f(\mathbf{x}_{\text{sub}}) = y] \right] \tag{10}$$

where $\mathbf{x}_{\text{sub}} \sim \text{RA}(\mathbf{x}_{1:t_0}, k)$ is to randomly sample k time points without replacement to construct the subseries $\mathbf{x}_{\text{sub}}$, and ablate all other points. $f_{\text{RA}}(\mathbf{x}_{1:t_0})$ returns the label that $f(\cdot)$ is most likely to classify $\mathbf{x}_{\text{sub}}$ as. $\hat{y} = f_{\text{RA}}(\mathbf{x}_{1:t_0})$ is provably robust if the top-1 probability score satisfies

$$\Pr_{\mathbf{x}_{\text{sub}} \sim \text{Sample}(\mathbf{x}_{1:t_0}, k)} [f(\mathbf{x}_{\text{sub}}) = \hat{y}] > \frac{3}{2} - \frac{\binom{t_0 - L_{\text{adv}}}{k}}{\binom{t_0}{k}} \tag{11}$$

Compare MIA to DS and RA. The base classifier of DS and RA are making predictions on subseries, instead of normal series, implying that the pretrained model would perform poorly on these two defenses. Therefore, to apply these two defenses, we need to train the base classifiers from scratch on the subseries. In stark contrast, MIA is a plug-and-play framework that can be applied to TSF/TSC pretrained models. The major cost of MIA is to train the imputation model. Remarkably, MIA imputation model can be trained in an unsupervised manner, saving us from labeling the data. Furthermore, our experimental results consistently demonstrate that MIA outperforms DS and RA on the certified accuracy, as further discussed in Section 5.

5 Experiments

Experimental setup. We evaluate MIA on both TSC and TSF datasets. TSF includes Exchange Rate, Traffic and UCI Electricity (Alexandrov et al., 2019), and TSC datasets include DistalPhalanxTW, MiddlePhalanxTW ProximalPhalanx (Ismail Fawaz et al., 2019a). We use MLP-Mixer (Tolstikhin et al., 2021), MLP and LSTM (Hochreiter & Schmidhuber, 1997) as the pretrained model. We emphasize that, our experiments are conducted on the clean trainsets, following the common setting in the field of certified adversarial defenses (Yoon et al., 2022; Li et al., 2020; Cohen et al., 2019; Chiang et al., 2020; Zhang et al., 2019). MLP-Mixer is set as the default pretrained model/base model for MIA, DS and RA. We use $\Delta = 1.5$ unless otherwise specified. All our experiments are conducted on CPU (16 Intel(R) Xeon(R) Gold 5222 CPU @ 3.80GHz) and GPU (one NVIDIA RTX 2080 Ti). More details about our experimental setups are omitted to Appendix.

Evaluation metrics. For TSC, we evaluate the defense by the *certified accuracy* (CA) to the temporally-localized perturbation of L_{adv}, defined by the fraction of the test samples that are correctly classified and certifiably robust to the perturbation. For TSC, we evaluate the defense by three metrics: *forecasting rate* (FR), *mean square error* (MSE) and *mean absolute error* (MAE)[4]. FR is the fraction of the time points on which MIA outputs the forecast instead of **Abstain**. MSE/MAE measures the mean square error/mean absolute error between forecasts (**Abstain** are excluded) and groundtruth.[5]

5.1 Comparison to Peer Methods

Comparison on TSC. Table 1 reports the certified accuracy of three methods in defending temporally-localized perturbations. The pretrained/base model architectures of three defenses are all MLP-Mixer. An interesting observation is that the certified accuracy of DS and RA keeps constant to different L_{adv}. The reason is that, the probability score of DS/RA models often concentrates on a single class, causing most classifications (including both (correct and wrong classifications) of DS

[4]We omit the evaluation of MAE to Appendix.

[5]In our experiments, $L_{\text{mask}} = c\%$ or $L_{\text{adv}} = c\%$ or $k = c\%$ refer to $c\% \cdot t_0$.

Table 2: (Exchange) Comparison among three certified defenses on TSF dataset.

Metric	Defense	1 %	2 %	3 %	4 %	5 %	6 %	7 %	8 %	9 %	10 %
FR (%)	MIA ($L_{\text{mask}} = 10\%$)	**82.2**	**82.2**	**83.2**	**82.2**	**82.2**	**81.2**	**81.2**	**81.2**	**80.2**	**79.2**
	DS ($k = 10\%$)	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8
	RA ($k = 10\%$)	16.8	16.8	16.8	16.8	16.8	16.8	16.8	16.8	16.8	16.8
	MIA ($L_{\text{mask}} = 15\%$)	**64.4**	**71.3**	**69.3**	**69.3**	**71.3**	**68.3**	**69.3**	**71.3**	**62.4**	**65.3**
	DS ($k = 15\%$)	20.8	20.8	20.8	14.9	14.9	11.9	11.9	11.9	11.9	10.9
	RA ($k = 15\%$)	23.8	23.8	23.8	23.8	23.8	23.8	23.8	23.8	23.8	23.8
MSE	MIA ($L_{\text{mask}} = 10\%$)	**0.143**	**0.141**	**0.145**	**0.141**	**0.141**	**0.139**	**0.137**	**0.137**	**0.135**	**0.134**
	DS ($k = 10\%$)	0.192	0.192	0.192	0.192	0.192	0.192	0.192	0.192	0.192	0.192
	RA ($k = 10\%$)	0.202	0.202	0.202	0.202	0.202	0.202	0.202	0.202	0.202	0.202
	MIA ($L_{\text{mask}} = 15\%$)	**0.126**	**0.134**	**0.130**	0.132	0.137	0.132	0.129	0.130	0.123	0.125
	DS ($k = 15\%$)	0.144	0.144	0.144	**0.065**	**0.065**	**0.069**	**0.069**	**0.069**	**0.069**	**0.060**
	RA ($k = 15\%$)	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248

Table 3: Comparison of the inference time (millisecond) of three defenses on TSC datasets.

Defense \ Model	FCN			MLP-Mixer			MLP			ResNet-18		
	2%	5%	10%	2%	5%	10%	2%	5%	10%	2%	5%	10%
MIA ($L_{\text{mask}} = 10\%$)	2.0	2.0	3.0	2.7	2.7	4.6	1.7	1.7	2.7	3.8	3.8	5.3
DS ($k = 10\%$)	0.6	0.6	0.6	2.0	2.0	2.0	0.3	0.3	0.3	2.6	2.6	2.6
RA ($k = 10\%$)	25.2	25.2	25.2	259.7	259.7	259.7	0.8	0.8	0.8	130.3	130.3	130.3
MIA ($L_{\text{mask}} = 15\%$)	2.0	2.0	2.0	2.7	2.7	2.7	1.7	1.7	1.7	3.8	3.8	3.8
DS ($k = 15\%$)	0.6	0.6	0.6	1.9	1.9	1.9	0.3	0.3	0.3	2.6	2.6	2.6
RA ($k = 15\%$)	25.3	25.3	25.3	260.6	260.6	260.6	0.8	0.8	0.8	130.4	130.4	130.4

Table 4: (Traffic) The performance of MIA on different pretrained models. (c_1 $c_2\%$) reports (MSE, FR%) of MIA. **Baseline** is MSE of the pretrained model without MIA. The lowest MSE and the highest FR for each pretrained model is shown in bold-face.

Model	Baseline	L_{mask}	$\Delta = 1.0$ 2%	$\Delta = 1.0$ 5%	$\Delta = 1.0$ 10%	$\Delta = 1.2$ 2%	$\Delta = 1.2$ 5%	$\Delta = 1.2$ 10%	$\Delta = 1.5$ 2%	$\Delta = 1.5$ 5%	$\Delta = 1.5$ 10%
MLP-Mixer	0.224	2%	**0.065** 72.3%			**0.072** 77.2%			0.144 89.1%		
		5%	0.067 75.2%	**0.068** 73.3%		0.079 **80.2%**	**0.075** 78.2%		**0.141** 90.1%	0.143 89.1%	
		10%	0.068 **77.2%**	0.069 **76.2%**	**0.066 69.3%**	0.079 **80.2%**	0.079 **80.2%**	**0.075 76.2%**	0.143 **91.1%**	**0.141 90.1%**	**0.139 88.1%**
GRU	0.243	2%	**0.067** 66.3%			**0.070** 73.3%			**0.143** 89.1%		
		5%	0.072 72.3%	**0.070** 68.3%		0.075 76.2%	**0.073** 74.3%		0.145 **91.1%**	**0.141** 88.1%	
		10%	0.070 **74.3%**	0.071 **72.3%**	**0.069 63.4%**	0.074 **77.2%**	0.074 **77.2%**	**0.066 70.3%**	0.145 **91.1%**	0.143 **90.1%**	**0.137 86.1%**
LSTM	0.229	2%	**0.068 66.3%**			**0.071** 76.2%			0.152 **91.1%**		
		5%	0.069 **66.3%**	**0.070** 65.3%		0.073 **77.2%**	**0.071** 76.2%		**0.149** 89.1%	**0.147** 88.1%	
		10%	0.070 65.3%	0.071 **66.3%**	**0.066 61.4%**	0.073 **77.2%**	0.073 **77.2%**	**0.064 72.3%**	0.150 89.1%	0.153 **90.1%**	**0.149 87.1%**
MLP	0.222	2%	**0.064** 67.3%			**0.064 72.3%**			0.148 90.1%		
		5%	0.067 **71.3%**	**0.064** 68.3%		**0.064 72.3%**	**0.064 72.3%**		0.148 **92.1%**	0.148 90.1%	
		10%	0.067 70.3%	0.067 **70.3%**	**0.063 66.3%**	**0.064 72.3%**	**0.064 72.3%**	**0.063 70.3%**	**0.146** 91.1%	**0.146 91.1%**	**0.144 89.1%**
ResNet18	0.248	2%	**0.074** 64.4%			0.087 75.2%			**0.149** 88.1%		
		5%	0.077 **68.3%**	**0.077** 65.3%		0.088 **79.2%**	0.089 **76.2%**		0.150 **89.1%**	**0.146 87.1%**	
		10%	0.077 67.3%	0.078 **67.3%**	**0.076 60.4%**	**0.086** 76.2%	**0.086 76.2%**	**0.083 73.3%**	0.150 88.1%	0.149 86.1%	**0.145 83.2%**

and RA are of high robustness. The results show that the certified accuracy of MIA is more than twice of DR and RA across different L_{adv}. The fundamental reason is that the pretrained model of MIA classifies the masked series, while the base model in RS/DS classifies the subseries. MIA can offer a higher certified accuracy with more information, as the masked series contains much more information ($t_0 - L_{\text{mask}}$ unmasked time points) than the subseries (k sampled time points).

Comparison on TSF. Table 2 reports FR and MSE of three defenses on Exchange, where the model predicts the next 30 values. Here we utilize discretization technique to make the TSF task feasible to DS and RA. The table shows that MIA offers a significantly higher FR than DS and RA, implying that MIA return the forecasting results much more frequently than other two defenses. The reason for the superior FR is same as TSC. We also observe that DS ($k = 15\%$) achieves a lower MSE than MIA at $L_{\text{adv}} \geq 4\%$, which partially owing to its low FR. Since a low FR implies that the aggregation step of MIA filters a large portion of distrustful forecasts, reducing the difficulty of achieving lower MSE for the remained forecasts. Based on the results, MIA is better than other two defenses when we jointly consider FR and MSE.

Comparison on inference time. Table 3 compares the inference time of three defenses, which is averaged among three TSC datasets. We observe that the inference time of MIA is larger than DS, but significantly smaller than RA. Specifically, the larger inference time than DS is owing to the additional cost in running the imputation model of MIA. The seriously large inference time of RA is because of

Table 5: (Electricity) (c_1 $c_2\%$) report (MSE $\mathrm{FR}\%$) of MIA on different pretrained models.

Model	Baseline	L_{mask}	$\Delta = 1.0$ 2%	$\Delta = 1.0$ 5%	$\Delta = 1.0$ 10%	$\Delta = 1.2$ 2%	$\Delta = 1.2$ 5%	$\Delta = 1.2$ 10%	$\Delta = 1.5$ 2%	$\Delta = 1.5$ 5%	$\Delta = 1.5$ 10%
MLP-Mixer	0.388	2%	**0.093** 66.3%			0.094 64.4%			0.136 77.2%		
		5%	0.095 **69.3%**	**0.093** **68.3%**		0.096 **67.3%**	0.094 **66.3%**		0.134 **78.2%**	0.134 **78.2%**	
		10%	0.095 67.3%	0.095 67.3%	**0.089** **55.4%**	**0.091** 64.4%	**0.091** 64.4%	**0.089** **59.4%**	**0.134** 77.2%	**0.134** 77.2%	**0.116** **66.3%**
GRU	0.420	2%	**0.099** 62.4%			0.086 60.4%			**0.135** 68.3%		
		5%	0.100 64.4%	0.101 **63.4%**		0.087 **62.4%**	**0.087** **62.4%**		0.139 **69.3%**	**0.139** **69.3%**	
		10%	0.102 **65.3%**	**0.099** 62.4%	**0.103** **49.5%**	**0.085** 61.4%	0.087 61.4%	**0.086** **53.5%**	0.139 **69.3%**	**0.139** **69.3%**	**0.120** **64.4%**
LSTM	0.438	2%	0.100 64.4%			**0.082** 52.5%			0.142 70.3%		
		5%	**0.100** **66.3%**	0.101 **65.3%**		0.092 55.4%	**0.092** **55.4%**		**0.140** **71.3%**	**0.142** **70.3%**	
		10%	**0.100** **66.3%**	**0.100** 64.4%	**0.110** **51.5%**	0.091 **56.4%**	**0.092** **55.4%**	**0.094** **48.5%**	**0.140** **71.3%**	0.142 **70.3%**	**0.119** **59.4%**
MLP	0.402	2%	0.106 73.3%			**0.082** 61.4%			0.136 74.3%		
		5%	0.108 **77.2%**	0.105 **75.2%**		0.087 **65.3%**	**0.085** **63.4%**		**0.131** **77.2%**	0.132 **76.2%**	
		10%	**0.105** 75.2%	**0.105** **75.2%**	**0.098** **60.4%**	0.087 61.4%	0.087 61.4%	**0.084** **55.4%**	**0.131** **77.2%**	**0.126** **76.2%**	**0.108** **66.3%**
ResNet18	0.554	2%	0.093 51.5%			0.080 59.4%			**0.136** 66.3%		
		5%	**0.089** 51.5%	**0.089** 50.5%		0.080 **60.4%**	**0.080** **60.4%**		0.140 **68.3%**	0.140 **68.3%**	
		10%	0.094 **52.5%**	0.093 **51.5%**	**0.092** **43.6%**	**0.079** 56.4%	0.081 58.4%	**0.076** **54.5%**	0.141 67.3%	**0.136** 66.3%	**0.121** **61.4%**

Table 6: Comparison of different training algorithms on 3 TSC datasets.

Model	Training	L_{mask}	DistalPhalanxTW 5%	DistalPhalanxTW 10%	DistalPhalanxTW 15%	MiddlePhalanxTW 5%	MiddlePhalanxTW 10%	MiddlePhalanxTW 15%	ProximalPhalanxTW 5%	ProximalPhalanxTW 10%	ProximalPhalanxTW 15%
MLP-Mixer	Random	5%	62.4%			52.5%			65.3%		
		10%	61.4%	58.4%		53.5%	50.5%		73.3%	71.3%	
		15%	59.4%	59.4%	58.4%	59.4%	54.5%	49.5%	68.3%	67.3%	61.4%
	Masked	5%	**65.3%**			**64.4%**			**72.3%**		
		10%	**66.3%**	**63.4%**		**67.3%**	**60.4%**		**76.2%**	**74.3%**	
		15%	**64.4%**	**64.4%**	**60.4%**	**66.3%**	**62.4%**	**59.4%**	**76.2%**	**75.2%**	**74.3%**
FCN	Random	5%	63.4%			53.5%			68.3%		
		10%	66.3%	63.4%		52.5%	52.5%		70.3%	67.3%	
		15%	66.3%	65.3%	65.3%	57.4%	57.4%	56.4%	66.3%	66.3%	65.3%
	Masked	5%	**70.3%**			**64.4%**			**75.2%**		
		10%	**70.3%**	**69.3%**		**66.3%**	**65.3%**		**73.3%**	**71.3%**	
		15%	**70.3%**	**67.3%**	**66.3%**	**65.3%**	**63.4%**	**63.4%**	**75.2%**	**73.3%**	**73.3%**
MLP	Random	5%	62.4%			65.3%			72.3%		
		10%	62.4%	60.4%		59.4%	54.5%		76.2%	73.3%	
		15%	63.4%	61.4%	61.4%	61.4%	55.4%	55.4%	71.3%	71.3%	68.3%
	Masked	5%	**64.4%**			**69.3%**			**79.2%**		
		10%	**65.3%**	**64.4%**		**70.3%**	**67.3%**		**79.2%**	**78.2%**	
		15%	**66.3%**	**63.4%**	**63.4%**	**69.3%**	**66.3%**	**66.3%**	**78.2%**	**78.2%**	**77.2%**
ResNet-18	Random	5%	61.4%			57.4%			65.3%		
		10%	59.4%	57.4%		57.4%	56.4%		74.3%	67.3%	
		15%	58.4%	56.4%	55.4%	58.4%	58.4%	58.4%	74.3%	73.3%	66.3%
	Masked	5%	**65.3%**			**67.3%**			**78.2%**		
		10%	**66.3%**	**64.4%**		**63.4%**	**63.4%**		**78.2%**	**76.2%**	
		15%	**64.4%**	**61.4%**	**59.4%**	**65.3%**	**62.4%**	**62.4%**	**78.2%**	**78.2%**	**78.2%**

its confidence interval estimation for the probability score, which requires a large number of samples of noisy classifications[6]. We also observe that the inference time of MIA increases with L_{adv}, and decreases with L_{mask}, since the number of masked series equals $\lfloor (t_0 - 1)/(L_{\text{mask}} - L_{\text{adv}} + 1) \rfloor + 1$. We omit the inference time analysis on TSF datasets to Appendix.

5.2 Analysis of MIA on Different Pretrained Models

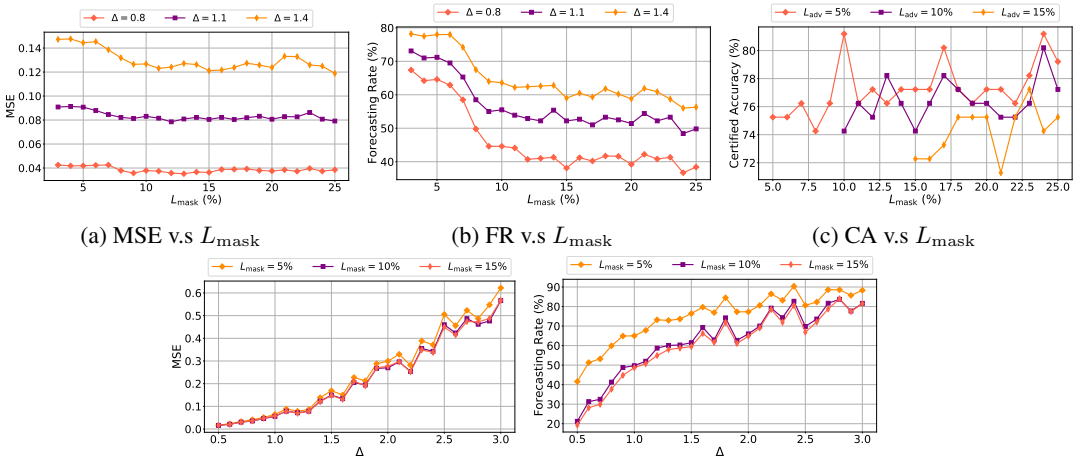
Table 4 and Table 5 report the performance of MIA on different pretrained models, where the model forecasts the next 24 points[7]. We observe that MIA ($\Delta = 1.0, 1.5$) consistently lowers MSE as compared to that of the original pretrained models, suggesting MIA could also be an effective plugin for performance improvement. Specifically, MIA improves the forecasting performance in the way of filtering these distrustful forecasts, sacrificing the availability (the decrease of FR) for lower MSE as well as certified robustness, which is a common trade-off in the field of certified defenses (Cohen et al., 2019; Levine & Feizi, 2020a;b; Liu et al., 2021; Han et al., 2021). We can control the trade-off between the forecasting performance and FR via modifying Δ, as the decrease of Δ can further reduce MSE at the cost of FR.

5.3 Analysis on Imputation Model of MIA and Ablation Study

Impact of training algorithm. Table 6 compares our masked training to random training, which trains the imputation model on the randomly masked series. Through extensive comparisons on different imputation model architectures and datasets, we convince that masked training consistently

[6]Following the official implementation of RA (Levine & Feizi, 2020b), we take $100,000$ samples for the confidence interval estimation.

[7]We omit our results on Exchange dataset to Appendix due to the space limitation.



(a) MSE v.s L_{mask} (b) FR v.s L_{mask} (c) CA v.s L_{mask}

(d) MSE v.s Δ (e) FR v.s Δ

Figure 2: **Top**: impact of L_{mask} on TSC dataset ProximalPhalanxTW. **Bottom**: impact of Δ on TSF dataset Traffic ($L_{\text{adv}} = 3\%$).

Table 7: Comparison of four imputation models. The best results are shown in bold-face.

Model	L_{mask}	Electricity 2%	Electricity 5%	Electricity 10%	Exchange 2%	Exchange 5%	Exchange 10%	Traffic 2%	Traffic 5%	Traffic 10%
MLP-Mixer	2%	0.136 **77.2%**			0.158 **87.1%**			0.144 **89.1%**		
	5%	0.134 **78.2%**	0.134 **78.2%**		0.144 **83.2%**	0.138 **81.2%**		0.141 **90.1%**	0.143 **89.1%**	
	10%	0.134 **77.2%**	0.134 **77.2%**	0.116 **66.3%**	0.141 **82.2%**	0.141 **82.2%**	0.134 **79.2%**	0.143 **91.1%**	0.141 **90.1%**	0.139 **88.1%**
BRITS	2%	0.118 65.3%			0.118 40.6%			0.142 87.1%		
	5%	0.110 71.3%	0.099 59.4%		0.118 49.5%	0.098 15.8%		0.136 84.2%	0.130 75.2%	
	10%	0.099 61.4%	0.100 60.4%	0.094 45.5%	0.207 10.9%	0.277 1.0%	0.000 0.0%	0.128 79.2%	0.129 77.2%	0.123 73.3%
SAITS	2%	0.101 58.4%			**0.095** 25.7%			**0.120** 67.3%		
	5%	**0.091** 53.5%	**0.090** 52.5%		0.137 43.6%	0.101 27.7%		0.128 73.3%	**0.121** 68.3%	
	10%	0.104 56.4%	0.103 54.5%	0.094 44.6%	0.070 7.9%	0.048 6.9%	**0.001** 4.0%	0.125 67.3%	0.132 69.3%	0.124 59.4%
Transformer	2%	**0.085** 52.5%			0.145 23.8%			0.130 68.3%		
	5%	0.110 52.5%	0.113 42.6%		**0.072** 6.9%	**0.001** 2.0%		**0.127** 74.3%	0.123 65.3%	
	10%	**0.096** 60.4%	**0.093** 58.4%	**0.093** 34.7%	**0.029** 8.9%	**0.036** 7.9%	0.001 5.0%	**0.124** 75.2%	**0.127** 74.3%	**0.123** 61.4%

outperforms random training for MIA by a non-trivial gap. The gap becomes even larger at larger L_{adv}. The results suggest that masked training is suitable for MIA imputation model.

Impact of architecture of imputation models. Table 7 reports the performances of different imputation model architectures on MIA. Our results show that MLP-Mixer can offer higher FR and lower MSE simultaneously, suggesting MLP-Mixer is intrinsically more robust than other models.

Impact of L_{mask}. Fig. 2a, 2b show: 1) MSE roughly keeps constant w.r.t. L_{mask}, because the discretization technique can diminish the difference between the forecasts that are close to each other. 2) FR decreases with the increase of L_{mask}, because our imputation quality decreases with L_{mask}, making it harder to reach agreement unanimously. Fig. 2c show that the impact of L_{mask} on CA is not significant. Although the increase of L_{mask} reduces our imputation quality to decrease FR, it also reduce the number of masked series, benefiting the agreement.

Impact of Δ. Fig. 2d, 2e report the impact of Δ. As Δ increases, we observe that MSE and FR increase, validating our statement about Δ in Section 4.1.

6 Conclusion

In this paper, we propose the first framework for time-series models to certifiably defend against ℓ_0-norm perturbations. Notably, MIA is a plug-and-play defense, which can be easily applied to any TSF/TSC pretrained model. The only requirement of deploying MIA is to train an imputation model, which has been extensively explored in this work. Moreover, our extensive experiments validate the effectiveness of MIA. We expect our work can inspire more studies on the ℓ_0-norm robustness for time-series models. Interesting future works include applying MIA to probabilistic models.

References

Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, et al. Gluonts: Probabilistic time series models in python. *arXiv preprint arXiv:1906.05264*, 2019.

Anish Athalye and Nicholas Carlini. On the robustness of the cvpr 2018 white-box adversarial example defenses. *arXiv preprint arXiv:1804.03286*, 2018.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. Brits: Bidirectional recurrent imputation for time series. In *NeurIPS*, 2018.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pp. 3–14, 2017.

Ping-yeh Chiang, Michael Curry, Ahmed Abdelkader, Aounon Kumar, John Dickerson, and Tom Goldstein. Detection as regression: Certified object detection with median smoothing. In *NeurIPS*, 2020.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, 2014.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *ICML*, 2017.

Charles J Clopper and Egon S Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 1934.

Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.

Jerome T Connor, R Douglas Martin, and Les E Atlas. Recurrent neural networks and robust time series prediction. *IEEE transactions on neural networks*, 1994.

Raphaël Dang-Nhu, Gagandeep Singh, Pavol Bielik, and Martin Vechev. Adversarial attacks on probabilistic autoregressive forecasting models. In *ICML*, 2020.

Emmanuel de Bézenac, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, and Tim Januschowski. Normalizing kalman filters for multivariate time series analysis. In *Neural Information Processing Systems*, 2020.

Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. *AAAI*, 2021.

Daizong Ding, Mi Zhang, Yuanmin Huang, Xudong Pan, Fuli Feng, Erling Jiang, and Min Yang. Towards backdoor attack on deep learning based time series classification. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pp. 1274–1287. IEEE, 2022.

Wenjie Du, David Côté, and Yan Liu. Saits: Self-attention-based imputation for time series. *arXiv preprint arXiv:2202.08516*, 2022.

Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. Gp-vae: Deep probabilistic time series imputation. In *International conference on artificial intelligence and statistics*, 2020.

Sarah Gelper, Roland Fried, and Christophe Croux. Robust forecasting with exponential and holt–winters smoothing. *Journal of forecasting*, 2010.

Tian Guo, Zhao Xu, Xin Yao, Haifeng Chen, Karl Aberer, and Koichi Funaya. Robust online time series prediction with recurrent neural networks. In *DSAA*. Ieee, 2016.

Husheng Han, Kaidi Xu, Xing Hu, Xiaobing Chen, Ling Liang, Zidong Du, Qi Guo, Yanzhi Wang, and Yunji Chen. Scalecert: Scalable certified defense against adversarial patches with sparse superficial layers. In *NeurIPS*, 2021.

Xintian Han, Yuxuan Hu, Luca Foschini, Larry Chinitz, Lior Jankelson, and Rajesh Ranganath. Deep learning models for electrocardiograms are susceptible to adversarial attack. *Nature medicine*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Söderström. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. *KDD*, 2018.

Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019a.

Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 2019b.

J. Ke, H. Zheng, H. Yang, and X. M. Chen. Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. *Transportation research part C: Emerging technologies*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

Alexander Levine and Soheil Feizi. (de) randomized smoothing for certifiable defense against patch attacks. In *NeurIPS*, 2020a.

Alexander Levine and Soheil Feizi. Robustness certificates for sparse adversarial attacks by randomized ablation. In *AAAI*, 2020b.

Dan Li, Dacheng Chen, Lei Shi, Baihong Jin, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *ICANN*, 2019.

Hui Li, Yunpeng Cui, Shuo Wang, Juan Liu, Jinyuan Qin, and Yilin Yang. Multivariate financial time-series prediction with certified robustness. *IEEE Access*, 2020.

Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei. Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding. *KDD*, 2021.

Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Pointguard: Provably robust 3d point cloud classification. In *CVPR*, 2021.

Yonghong Luo, Ying Zhang, Xiangrui Cai, and Xiaojie Yuan. E2gan: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of the 28th international joint conference on artificial intelligence*, 2019.

Michael McCoyd, Won Park, Steven Chen, Neil Shah, Ryan Roggenkemper, Minjune Hwang, Jason Xinyu Liu, and David Wagner. Minority reports defense: Defending against adversarial patches. In *International Conference on Applied Cryptography and Network Security*, 2020.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. In *ICLR*, 2017.

Steffen Moritz and Thomas Bartz-Beielstein. imputets: time series missing value imputation in r. *R J.*, pp. 207, 2017.

Izaskun Oregi, Javier Del Ser, Aritz Perez, and Jose A Lozano. Adversarial sample crafting for time series classification with elastic similarity measures. In *International Symposium on Intelligent and Distributed Computing*, pp. 26–39. Springer, 2018.

Daehyung Park, Yuuna Hoshi, and Charles C. Kemp. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *RA-L*, 2018.

Lukas Ruff, Nico Görnitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Robert A. Vandermeulen, Alexander Binder, Emmanuel Müller, and M. Kloft. Deep one-class classification. In *ICML*, 2018.

David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. *NeurIPS*, 2019.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, pp. 1181–1191, 2020.

Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J Zico Kolter. Denoised smoothing: A provable defense for pretrained classifiers. In *NeurIPS*, 2020.

T. Schlegl, Philipp Seeböck, S. Waldstein, G. Langs, and U. Schmidt-Erfurth. f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Med. Image Anal.*, 2019.

B. Schölkopf, John C. Platt, J. Shawe-Taylor, Alex Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 2001.

Shahroz Tariq, Sangyup Lee, Youjin Shin, Myeong Shin Lee, Okchul Jung, Daewon Chung, and Simon S. Woo. Detecting anomalies in space using multivariate convolutional lstm with mixtures of probabilistic pca. *KDD*, 2019.

D. Tax and R. Duin. Support vector data description. *Mach. Learn.*, 2004.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. In *NeurIPS*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Eric Wong and J Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

Chong Xiang and Prateek Mittal. Patchguard++: Efficient provable attack detection against adversarial patches. *arXiv*, 2021.

Chong Xiang, Arjun Nitin Bhagoji, Vikash Sehwag, and Prateek Mittal. {PatchGuard}: A provably robust defense against adversarial patches via small receptive fields and masking. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2237–2254, 2021.

Chong Xiang, Saeed Mahloujifar, and Prateek Mittal. {PatchCleanser}: Certifiably robust defense against adversarial patches for any image classifier. In *31st USENIX Security Symposium (USENIX Security 22)*, pp. 2065–2082, 2022.

Haowen Xu, Wenxiao Chen, N. Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Y. Liu, Y. Zhao, Dan Pei, Yang Feng, Jian Jhen Chen, Zhaogang Wang, and Honglin Qiao. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. *WWW*, 2018.

Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *ICLR*, 2022.

Takehisa Yairi, Naoya Takeishi, Tetsuo Oda, Yuta Nakajima, Naoki Nishimura, and Noboru Takata. A data-driven health monitoring method for satellite housekeeping data based on probabilistic clustering and dimensionality reduction. *IEEE Trans. Aerosp. Electron. Syst.*, 2017.

Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *ICML*, 2017.

Wenbo Yang, Jidong Yuan, Xiaokang Wang, and Peixiang Zhao. Tsadv: Black-box adversarial attack on time series with local perturbations. *Engineering Applications of Artificial Intelligence*, 114: 105218, 2022.

Yun Yang and ChongJun Fan. Efficient and robust time series prediction model based on remd-mmlp with temporal-window. *Expert Systems with Applications*, 2022.

Zhongguo Yang, Han Li, Mingzhu Zhang, Jingbin Wang, and Chen Liu. A method for resisting adversarial attack on time series classification model in iot system. In *International Conference on Web Information Systems and Applications*, pp. 559–566. Springer, 2020.

TaeHo Yoon, Youngsuk Park, Ernest K Ryu, and Yuyang Wang. Robust probabilistic time series forecasting. In *AISTATS*, 2022.

Ceylan Yozgatligil, Sipan Aslan, Cem Iyigun, and Inci Batmaz. Comparison of missing value imputation methods in time series: the case of turkish meteorological data. *Theoretical and applied climatology*, 2013.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. *arXiv preprint arXiv:1906.06316*, 2019.

Zhanyuan Zhang, Benson Yuan, Michael McCoyd, and David Wagner. Clipped bagnet: Defending against sticker attacks with clipped bag-of-features. In *2020 IEEE Security and Privacy Workshops (SPW)*, pp. 55–61. IEEE, 2020.

Hang Zhao, Yujing Wang, Juanyong Duan, Congrui Huang, Defu Cao, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Multivariate time-series anomaly detection via graph attention network. *ICDM*, 2020.

Bin Zhou, Shenghua Liu, Bryan Hooi, Xueqi Cheng, and Jing Ye. Beatgan: Anomalous rhythm detection using adversarially generated time series. In *IJCAI*, 2019.

A Proof for Proposition 1

Proposition 2 (Robustness Certificate of MIA)**.** *The forecast/label (not **Abstain**) returned by Algorithm 1 cannot be changed by any temporally-localized perturbation whose ℓ_0 norm is no larger than L_{adv} (see proof in Appendix).*

Proof. Here we prove the robustness certificate for MIA (TSC). The proof for MIA (TSF) is analogous to this proof. Assume that the adversary has changed the classification result of MIA from y_1 to y_2 via the temporally-localized perturbation δ (ℓ_0 norm is L_{adv}). Then, we have:

$$f(\mathbf{x}_{1:t_0} \odot \mathbf{M}^{(0)}) = f(\mathbf{x}_{1:t_0} \odot \mathbf{M}^{(1)}) = \ldots = f(\mathbf{x}_{1:t_0} \odot \mathbf{M}^{(M)}) = y_1 \tag{12}$$

$$f((\mathbf{x}_{1:t_0} + \delta) \odot \mathbf{M}^{(0)}) = f((\mathbf{x}_{1:t_0} + \delta) \odot \mathbf{M}^{(1)}) = \ldots = f((\mathbf{x}_{1:t_0} + \delta) \odot \mathbf{M}^{(M)}) = y_2 \tag{13}$$

Without loss of generality, suppose the temporally-localized perturbation is located at $[t_{\text{adv}} + 1 : \min(t_{\text{adv}} + L_{\text{adv}}, t_0)]$ ($t_{adv} = 0, \ldots, t_0 - 1$), we prove that *there exists a mask $M^{(\hat{m})}, \hat{m} = \lfloor \frac{t_{\text{adv}}}{\alpha} \rfloor$ that can completely occlude it*. Specifically, we compare the starting/end point of the mask $\mathbf{M}^{(\hat{m})}$ and the perturbation δ. For the starting point, we have:

$$\underbrace{(\alpha \lfloor \frac{t_{\text{adv}}}{\alpha} \rfloor + 1)}_{\text{Mask}} - \underbrace{(t_{\text{adv}} + 1)}_{\text{Perturbation}} \leq 0 \tag{14}$$

In terms of the end points, we have:

$$\underbrace{(\alpha\lfloor\frac{t_{\text{adv}}}{\alpha}\rfloor+1+L_{\text{mask}})}_{\text{Mask}}-\underbrace{(t_{\text{adv}}+L_{\text{adv}})}_{\text{Perturbation}} \tag{15}$$

$$=\alpha\lfloor\frac{t_{\text{adv}}}{\alpha}\rfloor+(L_{\text{mask}}-L_{\text{adv}}+1)-t_{\text{adv}} \tag{16}$$

$$=\alpha(\lfloor\frac{t_{\text{adv}}}{\alpha}\rfloor+1)-t_{\text{adv}}\geq 0 \tag{17}$$

As $\mathrm{M}^{(\hat{m})}$ occludes δ, thus $(\mathbf{x}_{1:t_0})\odot\mathrm{M}^{(\hat{m})}=(\mathbf{x}_{1:t_0}+\delta)\odot\mathrm{M}^{(\hat{m})}\Rightarrow y_1=y_2$. Our proof is completed.

B Empirical Evaluation on Risk of Temporally-Localized Perturbations

To support our statement about the risk of temporally-localized perturbations, we specifically propose an algorithm for generating the temporally-localized perturbations. We then evaluate the attack performance visually.

B.1 Restate Definition of temporally-localized Perturbation

Definition 2 (Temporally-localized perturbation)**.** *Temporally-localized perturbation is to* perturb consecutive time points of $\mathbf{x}_{1:t_0}$ w.r.t. ℓ_0-norm constraint*. The perturbed series is:*

$$\begin{aligned}&\boldsymbol{x}_{1:t_0}+\delta,\quad \textit{subject to}\quad \|\delta\|_0\leq L_{\text{adv}}\\ =&\boldsymbol{x}_{1:t_0}+[0,\ldots,\delta_{t_{\text{adv}}+1},\ldots,\delta_{t_{\text{adv}}+L_{\text{adv}}},0,\ldots,0]\\ =&[x_1,\ldots,\underbrace{x_{t_{\text{adv}}+1}+\delta_{t_{\text{adv}}+L_{\text{adv}}},\ \ldots,\ x_{t_{\text{adv}}+L_{\text{adv}}}+\delta_{t_{\text{adv}}+1}}_{\textit{Perturbed subsequence}},\ \ldots,x_{t_0}]\end{aligned} \tag{18}$$

where $t_{\text{adv}}+1$ and L_{adv} are the starting point and the ℓ_0-norm of the temporally-localized perturbation δ.

B.2 Algorithm of Generating Temporally-Localized Perturbations.

The objective of our algorithm is to maximize MSE between the original forecasts and the perturbed forecasts, with respect to the ℓ_0-norm constraint. Specifically, given the forecasting model $f(\mathbf{x}_{1:t_0})\rightarrow\mathbf{x}_{t_0+1,t_0+\tau}$, our objective can be formulated as follows:

$$\arg\max_{\delta}|f(\mathbf{x}_{1:t_0}+\delta)-f(\mathbf{x}_{1:t_0})|_2^2 \tag{19}$$

where δ corresponds to the perturbation defined in Eq.(18). Actually, the problem of computing the temporally-localized perturbation can be decomposed into two sub-problems: **P1**) Search for the period $[t_{\text{adv}}+1,t_{\text{adv}}+L_{\text{adv}}]$ to perturb. **P2**) Fix the period $[t_{\text{adv}}+1,t_{\text{adv}}+L_{\text{adv}}]$, compute the value of the perturbation $\delta_{t_{\text{adv}}+1},\ldots,\delta_{t_{\text{adv}}+L_{\text{adv}}}$. Here solving **P2** is not hard. If we have determined $\delta_{t_{\text{adv}}+1},\ldots,\delta_{t_{\text{adv}}+L_{\text{adv}}}$, we can maximize the following loss to compute the perturbation values via projected gradient descent (PGD).

$$\max_{\delta_{t_{\text{adv}}+1},\ldots,\delta_{t_{\text{adv}}+L_{\text{adv}}}}|f(\mathbf{x}_{1:t_0}+\delta)-f(\mathbf{x}_{1:t_0})|_2^2 \tag{20}$$

Then the main challenge is to determined which period to perturb. Here we solve **P1** by enumerating all the possible perturbing positions $[t_{\text{adv}}+1:t_{\text{adv}}+L_{\text{adv}}],\ t_{\text{adv}}=0,\ldots,t_0-L_{\text{adv}}$ and compute the corresponding attacks. Finally, we return the one with the largest MSE loss among $t_0-L_{\text{adv}}+1$ perturbations. However, in practice we found that computing the values of perturbation (**P2**) w.r.t. to the fixed period is hard to converge, as the ℓ_2 norm of the temporally-localized perturbation will approach ∞. We believe that a perturbation attack with ∞ ℓ_2 norm is meaningless in practice. In the sake of practicality, we additionally consider ℓ_2 norm for the temporally-localized perturbations besides ℓ_0-norm constraint for the sub-problem **P2** , as follows:

$$\max_{\delta_{t_{\text{adv}}+1},\ldots,\delta_{t_{\text{adv}}+L_{\text{adv}}}}|f(\mathbf{x}_{1:t_0}+\delta)-f(\mathbf{x}_{1:t_0})|_2^2\text{subject to}\|\delta\|_2^2\leq\epsilon \tag{21}$$

where ϵ is the preset upper bound of the perturbation ℓ_2 norm.

Table 8: (Traffic) Evaluate MSE between the clean forecasts and the perturbed forecasts. The temporally-localized perturbations is generated subject to different ℓ_0-norm constraints ($L_{\text{adv}} = 2\%, 5\%, 10\%$) and ℓ_2-norm constraints ($1.0, 1.5, 2.0, 2.5, 3.0, 3.5$).

Model	L_{adv}	1.0	1.5	2.0	2.5	3.0	3.5
MLP	2%	1.614	1.675	1.731	1.547	1.482	1.469
	5%	1.437	1.375	1.295	1.288	1.472	1.590
	10%	1.318	1.204	1.137	1.418	1.686	1.865
MLP-Mixer	2%	0.156	0.156	0.188	0.283	0.400	0.425
	5%	0.154	0.502	0.322	0.148	0.148	0.148
	10%	0.071	0.321	0.282	0.201	0.220	0.150
LSTM	2%	0.059	0.132	0.152	0.422	0.212	0.176
	5%	0.323	0.503	0.646	0.630	0.852	0.929
	10%	0.166	0.290	0.652	0.803	1.301	1.753

B.3 Empirical Evaluation of Temporally-localized Perturbations.

Quantify the risk of temporally-localized perturbations. Table 8 quantifies the risk of temporally-localized perturbations via computing MSE between the clean forecasts and the perturbed forecasts w.r.t. the ℓ_0-norm (L_{adv}) and the ℓ_2-norm (ϵ) constraints. We observe that MLP-Mixer model provides the highest empirical robustness among three models, which partially explains why MLP-Mixer outperforms other models on MIA. In particular, we further compare MLP to MLP+MIA ($\delta = 1.5$) in Table 4 of the main paper. Specifically, MSE of MLP under temporally-localized perturbations ($\epsilon = 3.0$) is $5\% : 1.472,\ 10\% : 0.1.686$ while MLP+MIA is $5\% : 0.146,\ 10\% : 0.144$. MIA reduces the MSE to roughly one tenth of the original, which indicates that MIA can effectively prevent our forecasting results from being influenced by the temporally-localized perturbations.

Visualize the risk of temporally-localized perturbations. Fig. 3 illustratively shows the effect of temporally-localized perturbations on our forecasting results. We observe that temporally-localized perturbations of $L_{\text{atk}} = 10\%$ can significantly change our forecasts.

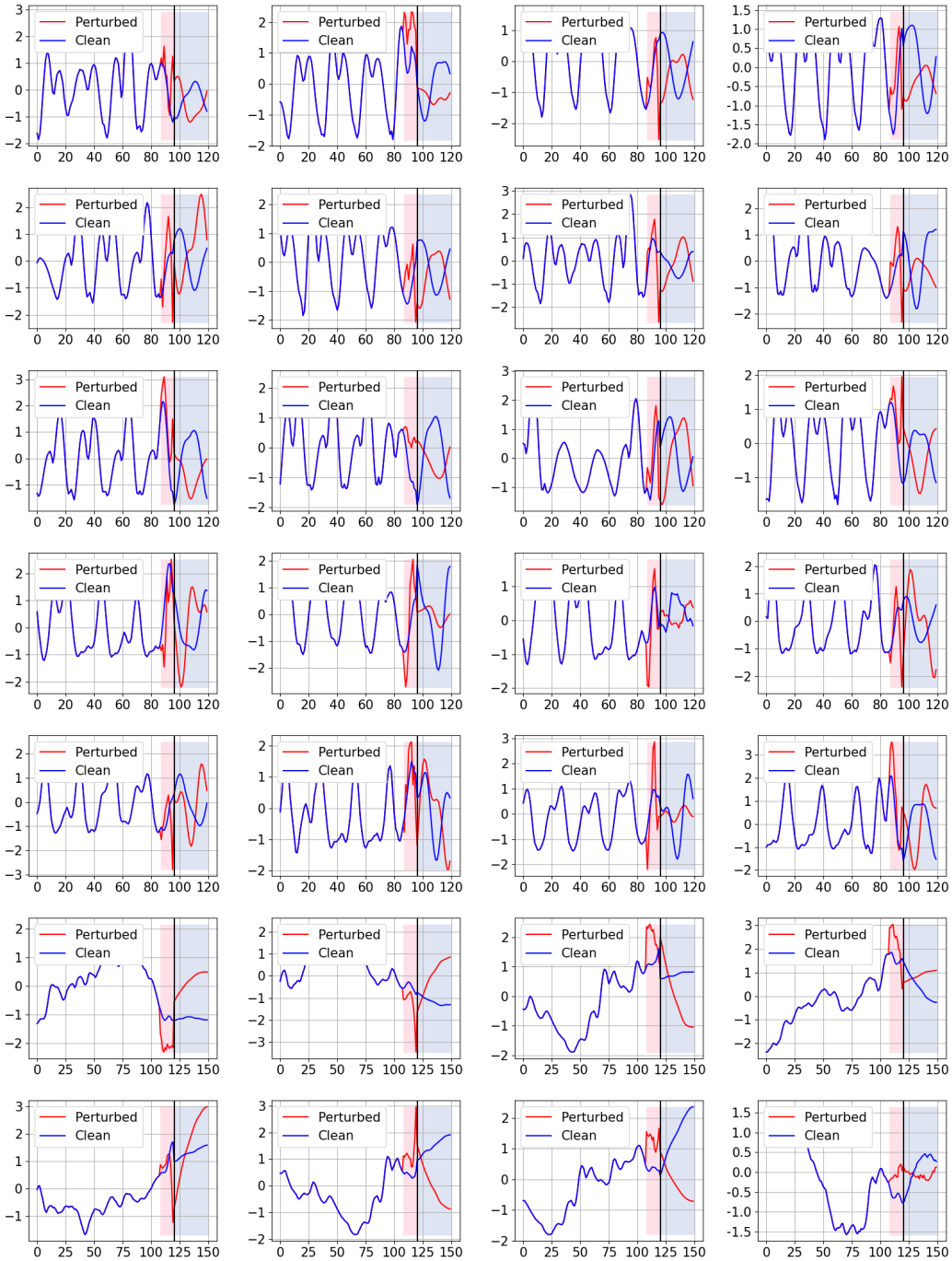


Figure 3: The effect of temporally-localized perturbations ($L_{\text{atk}} = 10\%$ and $\epsilon = 3.0$) on different datasets. Clean and Perturbed refer to the normal input series and the temporally-localized perturbations respectively. **Row 1, 2, 3**: Traffic. **Row 4, 5**: Electricity. **Row 6, 7**: Exchange rate. The red background denotes the position of the location of the perturbation. The blue background denotes the output series.

Table 9: Dataset information for TSF and TSC.

Dataset	Context length	Forecasting length	Number of classes
Electricity	96	24	N/A
Exchange	120	30	N/A
Traffic	96	24	N/A
DistalPhalanxTW	80	N/A	6
MiddlePhalanxTW	80	N/A	6
ProximalPhalanxTW	80	N/A	6

C Experimental Setups

C.1 Dataset Information

Table 9 shows the details of each dataset, including context length, forecasting length (for TSF datasets) and number of classes.

Traffic Hourly occupancy rate, between 0 and 1, of 963 San Francisco car lanes Salinas et al. (2019).

Electricity Hourly time series of the electricity consumption of 370 customers Salinas et al. (2019).

Exchange Daily exchange rate between 8 currencies Salinas et al. (2019).

DistalPhalanxTW, MiddlePhalanxTW, ProximalPhalanxTw [8] This series of 11 classification problems were created as part of Luke Davis's PhD titled "Predictive Modelling of Bone Ageing". They are designed to test the efficacy of hand and bone outline detection and whether these outlines could be helpful in bone age prediction. Note that these problems are aligned by subject, and hence can be treated as a multi-dimensional TSC problem. The final three bone classification problems, DistalPhalanxTW, MiddlePhalanxTW and ProximalPhalanxTW, involve predicting the Tanner-Whitehouse score (as labelled by a human expert) from the outline.

Data Pre-Processing We pre-process the input series with *scipy.signal.savgol_filter* with window length 15 and polyorder 5 on both training and testing datasets. Besides, we normalize each input series with its mean value and standard deviation. Mean value and standard deviation will be 0 and 1 respectively for each normalized input series. We use the instance normalization method on both trainsets and testsets.

D Introduction to Pretrained Models

Model architecture. We use the classical forecasting and classification models, MLP, MLP-Mixer Tolstikhin et al. (2021), GRU Cho et al. (2014), LSTM Hochreiter & Schmidhuber (1997), FCN Ismail Fawaz et al. (2019b) and ResNet-18 He et al. (2016) as the pretrained models. We show the architecture of these models in Table 10, 11, 12, 13.

Training. we uniformly adapt Adam optimizer Kingma & Ba (2014) with $lr = 0.0001, \beta = (0.9, 0.999), \epsilon = 10^{-8}, \text{weight_decay} = 0, \text{epochs} = 20$ for all the pretrained models.

E More Experimental Results

Comparison to peer methods on Mean Absolute Error (MAE). Table 15 compares MAE and FR of three defenses on Traffic. Analogous to the comparison (Table 2 in main paper) on MSE, MIA achieves both the lowest MAE and highest FR on $L_{\text{adv}} = 1\%, 2\%, 3\%$. DS outperforms MIA at $L_{\text{adv}} = 4\%, \ldots, 10\%$ for the great sacrifice on its FR.

[8] https://timeseriesclassification.com/description.php

Table 10: MLP structure

Input dim	Output dim	Activation
Length of sequence	96	LeakyReLU($\alpha = 0.2$)
96	96	LeakyReLU($\alpha = 0.2$)
96	96	LeakyReLU($\alpha = 0.2$)
96	96	LeakyReLU($\alpha = 0.2$)
96	Length of sequence	LeakyReLU($\alpha = 0.2$)

Table 11: LSTM/GRU structure

Number of layers	Hidden dim
4	32

Table 12: The structure of MLP-Mixer Block, and we stack 4 MLP-Mixer blocks to construct MLP-Mixer for forecasting and imputation.

Type	Input dim	Output dim	Activation
LayerNorm	96	96	/
Linear	128	128	GELU Hendrycks & Gimpel (2016)
Linear	128	128	GELU
LayerNorm	96	96	/
Linear	Length of sequence	Length of sequence	GELU
Linear	Length of sequence	Length of sequence	GELU

Table 13: The structure of Fully Convolutional Network (FCN) for TSC.

Layer	Input channel	Output channel	Kernel size
Conv1d	1	96	3
BatchNorm1d	96	96	N/A
ReLU	96	96	N/A
Conv1d	96	96	3
BatchNorm1d	96	96	N/A
ReLU	96	96	N/A
Conv1d	96	96	3
BatchNorm1d	96	96	N/A
ReLU	96	96	N/A
Conv1d	96	96	3
BatchNorm1d	96	96	N/A
ReLU	96	96	N/A
GlobalPooling	96	96	N/A
Linear	96	6	N/A

Table 14: Comparison of imputation quality (MSE between the imputed series and the original series) of different imputation models on imputing the masked series of different mask length $L_{\text{mask}} = 5\%, 10\%, 15\%, 20\%$. Fixing L_{mask}, we first construct $t_0 - L_{\text{mask}} + 1$ masked series of L_{mask} and compute the average imputation MSE over imputing these $t_0 - L_{\text{mask}} + 1$ masked series. Bold indicates the best among four generators.

Generator	Traffic				Electricity				Exchange			
	5%	10%	15%	20%	5%	10%	15%	20%	5%	10%	15%	20%
MLP-Mixer	**0.0007**	**0.0082**	**0.0170**	**0.0240**	**0.0033**	**0.0320**	**0.0458**	**0.0631**	**0.0002**	**0.0028**	**0.0104**	**0.0212**
SAITS	0.0652	0.1110	0.1786	0.2476	0.0878	0.1654	0.2687	0.3586	0.0191	0.0410	0.0631	0.0824
Transformer	0.0642	0.1183	0.1854	0.2629	0.0821	0.1836	0.2404	0.3271	0.0210	0.0389	0.0595	0.0787
BRITS	0.0159	0.0419	0.0656	0.0927	0.0410	0.1192	0.1613	0.2109	0.0099	0.0252	0.0516	0.0705

Evaluation of MAE on Traffic and Electricity. Table 16, Table 17 report MAE and FR of MIA on Traffic, Electricity and Exchange respectively, as a supplement to Table 4 and Table 5 in the main paper. We observe that MIA consistently reduces MAE of the pretrained models, similar to the results on MSE. Our comprehensive experiments suggest that MIA can effectively improve our forecasting quality in the way of filtering the unconfident forecasts.

Table 15: (Exchange) Comparison among three certified defenses on TSF dataset.

Metric	Defense	1 %	2 %	3 %	4 %	5 %	6 %	7 %	8 %	9 %	10 %
FR (%)	MIA ($L_{\text{mask}} = 10\%$)	**82.2**	**82.2**	**83.2**	**82.2**	**82.2**	**81.2**	**81.2**	**81.2**	**80.2**	**79.2**
	DS ($k = 10\%$)	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8
	RA ($k = 10\%$)	16.8	16.8	16.8	16.8	16.8	16.8	16.8	16.8	16.8	16.8
	MIA ($L_{\text{mask}} = 15\%$)	**64.4**	**71.3**	**69.3**	**69.3**	**71.3**	**68.3**	**69.3**	**71.3**	**62.4**	**65.3**
	DS ($k = 15\%$)	20.8	20.8	20.8	14.9	14.9	11.9	11.9	11.9	11.9	10.9
	RA ($k = 15\%$)	23.8	23.8	23.8	23.8	23.8	23.8	23.8	23.8	23.8	23.8
MAE	MIA ($L_{\text{mask}} = 10\%$)	**0.332**	**0.330**	**0.334**	**0.330**	**0.330**	**0.327**	**0.326**	**0.326**	**0.323**	**0.320**
	DS ($k = 10\%$)	0.408	0.408	0.408	0.408	0.408	0.408	0.408	0.408	0.408	0.408
	RA ($k = 10\%$)	0.413	0.413	0.413	0.413	0.413	0.413	0.413	0.413	0.413	0.413
	MIA ($L_{\text{mask}} = 15\%$)	**0.307**	**0.320**	**0.313**	0.316	0.322	0.317	0.311	0.314	0.301	0.306
	DS ($k = 15\%$)	0.320	0.320	0.320	**0.215**	**0.215**	**0.222**	**0.222**	**0.222**	**0.222**	**0.204**
	RA ($k = 15\%$)	0.446	0.446	0.446	0.446	0.446	0.446	0.446	0.446	0.446	0.446

Table 16: (Traffic) The performance of MIA on different pretrained models. (c_1 $c_2\%$) reports (MAE, FR%) of MIA. **Baseline** is MAE of the pretrained model without MIA. The lowest MAE and the highest FR for each pretrained model is shown in bold-face.

Model	Baseline	L_{mask}	$\Delta = 1.0$ 2%	$\Delta = 1.0$ 5%	$\Delta = 1.0$ 10%	$\Delta = 1.2$ 2%	$\Delta = 1.2$ 5%	$\Delta = 1.2$ 10%	$\Delta = 1.5$ 2%	$\Delta = 1.5$ 5%	$\Delta = 1.5$ 10%
MLP-Mixer	0.265	2%	**0.227** 72.3%			**0.213** 77.2%			0.340 89.1%		
		5%	0.231 75.2%	0.233 73.3%		0.225 **80.2%**	**0.218** 78.2%		**0.336** 90.1%	0.340 89.1%	
		10%	0.233 **77.2%**	**0.233** **76.2%**	**0.227** **69.3%**	0.225 **80.2%**	0.225 **80.2%**	**0.216** **76.2%**	0.338 **91.1%**	**0.336** **90.1%**	**0.333** **88.1%**
GRU	0.283	2%	**0.234** 66.3%			**0.210** 73.3%			**0.339** 89.1%		
		5%	0.242 72.3%	0.239 68.3%		0.219 76.2%	**0.214** 74.3%		0.340 **91.1%**	**0.337** 88.1%	
		10%	0.237 **74.3%**	**0.239** **72.3%**	**0.236** **63.4%**	0.217 **77.2%**	0.217 **77.2%**	**0.203** **70.3%**	0.340 **91.1%**	0.338 **90.1%**	**0.329** **86.1%**
LSTM	0.274	2%	**0.235** **66.3%**			**0.212** 76.2%			0.350 **91.1%**		
		5%	0.237 **66.3%**	**0.237** 65.3%		0.215 **77.2%**	**0.213** 76.2%		**0.345** 89.1%	**0.343** 88.1%	
		10%	0.237 65.3%	0.239 **66.3%**	**0.231** **61.4%**	0.215 **77.2%**	0.215 **77.2%**	**0.201** **72.3%**	0.347 89.1%	0.349 **90.1%**	**0.344** **87.1%**
MLP	0.268	2%	**0.226** 67.3%			**0.201** **72.3%**			0.344 90.1%		
		5%	0.231 **71.3%**	**0.226** 68.3%		**0.201** **72.3%**	**0.201** **72.3%**		0.343 **92.1%**	0.344 90.1%	
		10%	0.230 70.3%	0.230 **70.3%**	**0.225** **66.3%**	**0.201** **72.3%**	**0.201** **72.3%**	**0.199** **70.3%**	**0.341** 91.1%	**0.341** **91.1%**	**0.338** **89.1%**
ResNet18	0.290	2%	**0.244** 64.4%			0.238 75.2%			**0.343** 88.1%		
		5%	0.249 **68.3%**	**0.249** 65.3%		0.239 **79.2%**	0.241 **76.2%**		0.345 **89.1%**	**0.340** **87.1%**	
		10%	0.249 67.3%	0.251 **67.3%**	**0.246** **60.4%**	**0.235** 76.2%	**0.235** **76.2%**	**0.231** **73.3%**	0.345 88.1%	0.343 86.1%	**0.337** **83.2%**

Table 17: (Electricity) (c_1 $c_2\%$) report (MAE FR%) of MIA on different pretrained models.

Model	Baseline	L_{mask}	$\Delta = 1.0$ 2%	$\Delta = 1.0$ 5%	$\Delta = 1.0$ 10%	$\Delta = 1.2$ 2%	$\Delta = 1.2$ 5%	$\Delta = 1.2$ 10%	$\Delta = 1.5$ 2%	$\Delta = 1.5$ 5%	$\Delta = 1.5$ 10%
MLP-Mixer	0.430	2%	**0.260** 66.3%			0.262 64.4%			0.312 77.2%		
		5%	0.264 **69.3%**	**0.262** **68.3%**		0.266 **67.3%**	0.263 **66.3%**		0.308 **78.2%**	0.308 **78.2%**	
		10%	0.264 67.3%	0.264 67.3%	**0.257** **55.4%**	**0.259** 64.4%	**0.259** 64.4%	**0.255** **59.4%**	**0.308** 77.2%	**0.308** 77.2%	**0.287** **66.3%**
GRU	0.441	2%	**0.269** 62.4%			**0.247** 60.4%			**0.312** 68.3%		
		5%	0.273 64.4%	0.274 **63.4%**		0.250 **62.4%**	**0.250** **62.4%**		0.317 **69.3%**	**0.317** **69.3%**	
		10%	0.275 **65.3%**	**0.270** 62.4%	**0.282** **49.5%**	0.247 61.4%	0.251 61.4%	**0.250** **53.5%**	0.317 **69.3%**	**0.317** **69.3%**	**0.295** **64.4%**
LSTM	0.447	2%	**0.273** 64.4%			**0.243** 52.5%			0.319 70.3%		
		5%	0.273 **66.3%**	0.275 **65.3%**		0.257 55.4%	**0.257** **55.4%**		**0.316** **71.3%**	**0.319** **70.3%**	
		10%	0.273 **66.3%**	**0.272** 64.4%	**0.298** **51.5%**	0.257 **56.4%**	**0.257** **55.4%**	**0.262** **48.5%**	**0.316** **71.3%**	0.320 **70.3%**	**0.294** **59.4%**
MLP	0.421	2%	0.283 73.3%			**0.242** 61.4%			0.313 74.3%		
		5%	0.287 **77.2%**	0.283 **75.2%**		0.249 **65.3%**	**0.245** **63.4%**		**0.304** **77.2%**	0.306 **76.2%**	
		10%	**0.283** 75.2%	**0.283** **75.2%**	**0.273** **60.4%**	0.250 61.4%	0.250 61.4%	**0.246** **55.4%**	**0.304** **77.2%**	**0.299** **76.2%**	**0.280** **66.3%**
ResNet18	0.569	2%	0.256 51.5%			0.243 59.4%			**0.318** 66.3%		
		5%	**0.248** 51.5%	**0.248** 50.5%		0.242 **60.4%**	**0.242** **60.4%**		0.324 **68.3%**	0.324 **68.3%**	
		10%	0.257 **52.5%**	0.256 **51.5%**	**0.255** **43.6%**	**0.240** 56.4%	0.244 58.4%	**0.237** **54.5%**	0.323 67.3%	**0.318** 66.3%	**0.302** **61.4%**

Evaluation of MSE and MAE for MIA on Exchange. Table 18 and Table 19 evaluate MSE/MAE of MIA on exchange. We observe that, MIA moderately increase MSE/MAE of the pretrained models because of the information loss for the discretization technique. Specifically, we can see that MSE/MAE of the pretrained models are commonly much smaller than that of Traffic and Electricity because Exchange dataset is much simpler. The better forecasting performance implies that we might lose more information for the discretization technique. On Exchange, information loss plays a more conspicuous role than the filtering function of MIA, causing the increase of MSE/MAE. This suggests that discretization technique might lower the forecasting performance when the pretrained models are precise enough.

Analysis of training algorithm of imputation models. Table 20 reportS MSE and FR of masked training and random training on the TSF dataset Traffic. Similar to the comparison on TSC datasets (Table 6 in the main paper), our masked training consistently achieves lower MSE than random training on different imputation models.

Table 18: (Exchange) c_1 $c_2\%$ report MSE FR% of MIA at different L_{atk} and L_{def}.

Model	Baseline	L_{mask}	$\Delta = 1.0$ 2%	$\Delta = 1.0$ 5%	$\Delta = 1.0$ 10%	$\Delta = 1.2$ 2%	$\Delta = 1.2$ 5%	$\Delta = 1.2$ 10%	$\Delta = 1.5$ 2%	$\Delta = 1.5$ 5%	$\Delta = 1.5$ 10%
MLP-Mixer	0.030	2%	0.060 **88.1%**			0.100 **87.1%**			0.158 **87.1%**		
		5%	**0.055** 83.2%	**0.053** 80.2%		0.096 84.2%	0.096 **84.2%**		0.144 83.2%	**0.138** 81.2%	
		10%	0.056 83.2%	0.053 **81.2%**	**0.051 78.2%**	**0.095** 84.2%	**0.095 84.2%**	**0.088 80.2%**	**0.141** 82.2%	0.141 **82.2%**	**0.134 79.2%**
GRU	0.040	2%	0.062 **83.2%**			0.104 83.2%			0.152 **84.2%**		
		5%	**0.060** 79.2%	**0.058** 75.2%		0.105 85.1%	**0.099** 80.2%		**0.150** 83.2%	**0.140** 77.2%	
		10%	0.061 82.2%	0.060 **80.2%**	**0.058 68.3%**	**0.103 86.1%**	0.100 **84.2%**	**0.098 75.2%**	0.152 **84.2%**	0.152 **84.2%**	**0.142 78.2%**
LSTM	0.042	2%	**0.061 83.2%**			**0.098 87.1%**			0.162 **85.1%**		
		5%	0.062 82.2%	**0.058** 78.2%		0.099 86.1%	**0.098** 83.2%		**0.159** 83.2%	**0.152** 79.2%	
		10%	0.063 82.2%	0.062 **80.2%**	**0.055 75.2%**	0.099 86.1%	0.099 **85.1%**	**0.098 82.2%**	0.160 84.2%	0.160 **84.2%**	**0.155 82.2%**
MLP	0.658	2%	0.065 **27.7%**			0.110 41.6%			0.210 **56.4%**		
		5%	**0.057** 26.7%	0.072 19.8%		**0.105 44.6%**	**0.094** 27.7%		**0.209** 53.5%	0.231 40.6%	
		10%	**0.057** 26.7%	**0.057 26.7%**	**0.069 18.8%**	0.107 **44.6%**	0.106 **43.6%**	**0.105 35.6%**	0.213 51.5%	**0.216 52.5%**	**0.240 39.6%**
ResNet18	0.053	2%	0.060 78.2%			0.096 **83.2%**			0.163 **88.1%**		
		5%	**0.058 79.2%**	**0.058** 77.2%		0.093 81.2%	0.093 **80.2%**		**0.163** 87.1%	**0.157** 85.1%	
		10%	0.058 78.2%	0.060 **78.2%**	**0.059 72.3%**	**0.091** 78.2%	**0.092** 78.2%	**0.090 76.2%**	**0.163** 87.1%	0.163 **87.1%**	**0.159 85.1%**

Table 19: (Exchange) c_1 $c_2\%$ report MAE FR% of MIA at different L_{adv} and L_{mask}.

Model	Baseline	L_{mask}	$\Delta = 1.0$ 2%	$\Delta = 1.0$ 5%	$\Delta = 1.0$ 10%	$\Delta = 1.2$ 2%	$\Delta = 1.2$ 5%	$\Delta = 1.2$ 10%	$\Delta = 1.5$ 2%	$\Delta = 1.5$ 5%	$\Delta = 1.5$ 10%
MLP-Mixer	0.133	2%	0.209 **88.1%**			0.270 **87.1%**			0.349 **87.1%**		
		5%	**0.199** 83.2%	**0.194** 80.2%		0.264 84.2%	0.264 **84.2%**		0.333 83.2%	**0.326** 81.2%	
		10%	0.200 83.2%	0.195 **81.2%**	**0.190 78.2%**	**0.263** 84.2%	**0.263 84.2%**	**0.252 80.2%**	**0.330** 82.2%	0.330 **82.2%**	**0.320 79.2%**
GRU	0.152	2%	0.211 **83.2%**			**0.274** 83.2%			0.342 **84.2%**		
		5%	**0.206** 79.2%	**0.203** 75.2%		0.276 85.1%	**0.267** 80.2%		**0.339** 83.2%	**0.325** 77.2%	
		10%	0.210 82.2%	0.209 **80.2%**	**0.203 68.3%**	0.274 **86.1%**	0.270 **84.2%**	**0.263 75.2%**	0.342 **84.2%**	0.342 **84.2%**	**0.330 78.2%**
LSTM	0.159	2%	**0.209 83.2%**			**0.266 87.1%**			0.352 **85.1%**		
		5%	0.210 82.2%	**0.203** 78.2%		0.266 86.1%	**0.264** 83.2%		**0.348** 83.2%	**0.340** 79.2%	
		10%	0.213 82.2%	0.211 **80.2%**	**0.199 75.2%**	0.266 86.1%	0.266 **85.1%**	**0.263 82.2%**	0.350 84.2%	0.350 **84.2%**	**0.344 82.2%**
MLP	0.639	2%	0.213 **27.7%**			0.284 41.6%			0.409 **56.4%**		
		5%	**0.196** 26.7%	0.229 19.8%		**0.274 44.6%**	**0.260** 27.7%		**0.405** 53.5%	0.447 40.6%	
		10%	**0.196** 26.7%	**0.196 26.7%**	**0.227 18.8%**	0.277 **44.6%**	0.274 **43.6%**	**0.275 35.6%**	0.412 51.5%	**0.416 52.5%**	**0.458 39.6%**
ResNet18	0.181	2%	0.205 78.2%			0.263 **83.2%**			0.355 **88.1%**		
		5%	**0.201 79.2%**	**0.201** 77.2%		0.257 81.2%	0.257 **80.2%**		**0.354** 87.1%	**0.347** 85.1%	
		10%	0.202 78.2%	0.205 **78.2%**	**0.201 72.3%**	**0.254** 78.2%	**0.256** 78.2%	**0.253 76.2%**	**0.354** 87.1%	0.354 **87.1%**	**0.349 85.1%**

Table 20: (Traffic) Comparison between mask methods. c_1 $c_2\%$ report MSE TPR% of MIA across $L_{\text{def}} = 2\%, 5\%, 10\%$, $L_{\text{atk}} = 2\%, 5\%, 10\%$, $\Delta = 1.0, 1.2, 1.5$. c_1. MSE(c_1) reports the forecasting performance.

Model	Mask Method	L_{mask}	$\Delta = 1.0$ 2%	$\Delta = 1.0$ 5%	$\Delta = 1.0$ 10%	$\Delta = 1.2$ 2%	$\Delta = 1.2$ 5%	$\Delta = 1.2$ 10%	$\Delta = 1.5$ 2%	$\Delta = 1.5$ 5%	$\Delta = 1.5$ 10%
GRU	random	2%	**0.068** 53.5%			**0.071** 63.4%			**0.144** 76.2%		
		5%	**0.071** 42.6%	0.075 28.7%		**0.063** 50.5%	**0.057** 43.6%		**0.145** 73.3%	**0.135** 62.4%	
		10%	0.073 30.7%	**0.069** 26.7%	0.075 23.8%	**0.052** 49.5%	**0.048** 45.5%	**0.051** 40.6%	**0.140** 66.3%	**0.138** 64.4%	**0.132** 59.4%
	block	2%	0.074 **64.4%**			0.073 **66.3%**			0.148 **84.2%**		
		5%	0.073 **66.3%**	**0.075 61.4%**		0.073 **66.3%**	0.073 **65.3%**		0.148 **83.2%**	0.145 **80.2%**	
		10%	**0.073 64.4%**	0.074 **63.4%**	**0.070 57.4%**	0.074 **65.3%**	0.074 **64.4%**	0.065 **59.4%**	0.148 **84.2%**	0.148 **84.2%**	0.145 **79.2%**
LSTM	random	2%	**0.064** 55.4%			0.073 62.4%			0.144 81.2%		
		5%	**0.063** 37.6%	**0.050** 25.7%		**0.051** 49.5%	**0.055** 38.6%		**0.129** 67.3%	**0.130** 58.4%	
		10%	**0.065** 37.6%	**0.053** 27.7%	**0.049** 14.9%	**0.050** 48.5%	**0.053** 45.5%	**0.046** 30.7%	**0.134** 68.3%	**0.128** 64.4%	**0.120** 53.5%
	block	2%	0.071 **61.4%**			**0.069 68.3%**			**0.144 85.1%**		
		5%	0.067 **59.4%**	0.067 **56.4%**		0.069 **68.3%**	0.068 **66.3%**		0.144 **85.1%**	0.141 **82.2%**	
		10%	0.068 **60.4%**	0.067 **58.4%**	0.067 **56.4%**	0.069 **67.3%**	0.069 **66.3%**	0.067 **65.3%**	0.144 **85.1%**	0.144 **85.1%**	0.142 **83.2%**
MLP-Mixer	random	2%	**0.073** 54.5%			0.085 66.3%			**0.147** 75.2%		
		5%	**0.074** 44.6%	**0.067** 27.7%		**0.072** 59.4%	**0.061** 47.5%		**0.137** 74.3%	**0.142** 63.4%	
		10%	**0.072** 42.6%	**0.073** 40.6%	**0.059** 26.7%	**0.065** 57.4%	**0.067** 53.5%	**0.061** 43.6%	**0.134** 74.3%	**0.130** 70.3%	**0.130** 61.4%
	block	2%	0.079 **67.3%**			**0.085 74.3%**			0.151 **86.1%**		
		5%	0.076 **67.3%**	0.078 **61.4%**		0.083 **76.2%**	0.079 **72.3%**		0.151 **86.1%**	0.149 **83.2%**	
		10%	0.079 **63.4%**	0.077 **61.4%**	0.076 **55.4%**	0.081 **72.3%**	0.077 **71.3%**	0.078 **69.3%**	0.151 **86.1%**	0.150 **85.1%**	0.145 **82.2%**
MLP	random	2%	**0.069** 53.5%			0.074 61.4%			**0.145** 75.2%		
		5%	0.072 44.6%	0.071 37.6%		**0.059** 58.4%	**0.044** 47.5%		**0.138** 78.2%	**0.138** 67.3%	
		10%	**0.072** 42.6%	0.073 40.6%	0.077 34.7%	**0.061** 53.5%	**0.062** 52.5%	**0.055** 48.5%	**0.140** 78.2%	**0.140** 75.2%	**0.133** 67.3%
	block	2%	0.072 **60.4%**			**0.068 68.3%**			0.146 **84.2%**		
		5%	**0.071 61.4%**	**0.071 58.4%**		0.070 **66.3%**	0.070 **66.3%**		0.145 **86.1%**	0.147 **85.1%**	
		10%	0.073 **58.4%**	**0.072 59.4%**	**0.069 52.5%**	0.065 **66.3%**	0.065 **66.3%**	0.065 **62.4%**	0.146 **84.2%**	0.147 **85.1%**	0.144 **82.2%**
ResNet18	random	2%	**0.071** 57.4%			**0.083** 66.3%			**0.143** 82.2%		
		5%	**0.073** 33.7%	**0.072** 29.7%		**0.063** 54.5%	**0.057** 46.5%		**0.136** 62.4%	**0.139** 55.4%	
		10%	**0.067** 32.7%	**0.070** 27.7%	0.076 17.8%	**0.059** 49.5%	**0.061** 49.5%	**0.047** 37.6%	**0.143** 61.4%	**0.130** 53.5%	**0.117** 45.5%
	block	2%	0.073 **63.4%**			0.089 **71.3%**			0.146 **86.1%**		
		5%	0.074 **62.4%**	0.074 **62.4%**		0.085 **68.3%**	0.084 **67.3%**		0.147 **87.1%**	0.146 **86.1%**	
		10%	0.075 **60.4%**	0.072 **59.4%**	**0.070 57.4%**	0.087 **68.3%**	0.084 **69.3%**	0.076 **62.4%**	0.143 **84.2%**	0.143 **84.2%**	0.140 **82.2%**

Analysis of Different Pretrained Models on MIA. Table 21 reports the MAE of MIA with different pretrained models, as a supplement to Table 7 in the main paper. The results demonstrate that MLP-Mixer outperforms all the other defenses.

Table 21: Comparison of four imputation models. The best results are shown in bold-face.

Generator	Metric	L_{mask}	Electricity 2%	Electricity 5%	Electricity 10%	Exchange 2%	Exchange 5%	Exchange 10%	Traffic 2%	Traffic 5%	Traffic 10%
MLP-Mixer	MAE	2%	0.312 **77.2%**			0.349 **87.1%**			0.340 **89.1%**		
		5%	0.308 **78.2%**	0.308 **78.2%**		0.333 **83.2%**	0.326 **81.2%**		0.336 **90.1%**	0.340 **89.1%**	
		10%	0.308 **77.2%**	0.308 **77.2%**	0.287 **66.3%**	0.330 **82.2%**	0.330 **82.2%**	0.320 **79.2%**	0.338 **91.1%**	0.336 **90.1%**	0.333 **88.1%**
BRITS	MAE	2%	0.290 65.3%			0.299 40.6%			0.338 87.1%		
		5%	0.283 71.3%	0.264 59.4%		0.288 49.5%	0.246 15.8%		0.327 84.2%	0.320 75.2%	
		10%	0.271 61.4%	0.272 60.4%	0.265 45.5%	0.381 10.9%	0.526 1.0%	Inf 0.0%	0.315 79.2%	0.319 77.2%	0.312 73.3%
SAITS	MAE	2%	0.269 58.4%			**0.258** 25.7%			**0.307** 67.3%		
		5%	**0.261** 53.5%	**0.256** 52.5%		0.317 43.6%	0.264 27.7%		0.321 73.3%	**0.308** 68.3%	
		10%	0.274 56.4%	0.278 54.5%	**0.263** 44.6%	0.179 7.9%	**0.123** 6.9%	**0.035** 4.0%	0.316 67.3%	0.325 69.3%	0.315 59.4%
Transformer	MAE	2%	**0.253** 52.5%			0.329 23.8%			0.322 68.3%		
		5%	0.288 52.5%	0.291 42.6%		**0.164** 6.9%	**0.030** 2.0%		**0.318** 74.3%	0.312 65.3%	
		10%	**0.266** 60.4%	**0.263** 58.4%	0.274 34.7%	**0.112** 8.9%	0.135 7.9%	0.037 5.0%	**0.313** 75.2%	**0.317** 74.3%	**0.312** 61.4%

F Imputation Models.

For the imputation model architectures, we take SAITS, Transformer and BRITS and MLP-Mixer. Specifically, for SAITS, we use the code from [9]. For SAITS Du et al. (2022), we set $d_{\text{model}} = 32, n_{\text{layers}} = 2, d_{\text{inner}} = 16, n_{\text{head}} = 4, d_{\text{k}} = 8, d_{\text{v}} = 8$. For Transformer Vaswani et al. (2017), we set $d_{\text{model}} = 32, n_{\text{layers}} = 2, d_{\text{inner}} = 16, n_{\text{head}} = 4, d_{\text{k}} = 8, d_{\text{v}} = 8$. For BRITS Cao et al. (2018), we set $h_{\text{hidden}} = 32$. For MLP-Mixer Tolstikhin et al. (2021), we use the same structure as Table 12. In terms of training, we use optimizer Adam with $lr = 0.0001, \beta = (0.9, 0.999), \epsilon = 10^{-8}$, $\text{weight_decay} = 0$ and train the model for 30 epochs.

Comparison of the imputation quality on imputing masked series. Table 14 compares the imputation quality of different imputation models on Traffic. The imputation quality is quantified by the mean square error (MSE) between the imputed series and the original series. We observe that MLP-Mixer consistently outperforms other three models across different datasets and L_{adv}, indicating the superior imputation ability of MLP-Mixer architecture.

Visualize the imputation quality. Fig. 4 intuitively shows the imputation performance of different imputation models on dataset Traffic.

[9] https://github.com/WenjieDu/PyPOTS

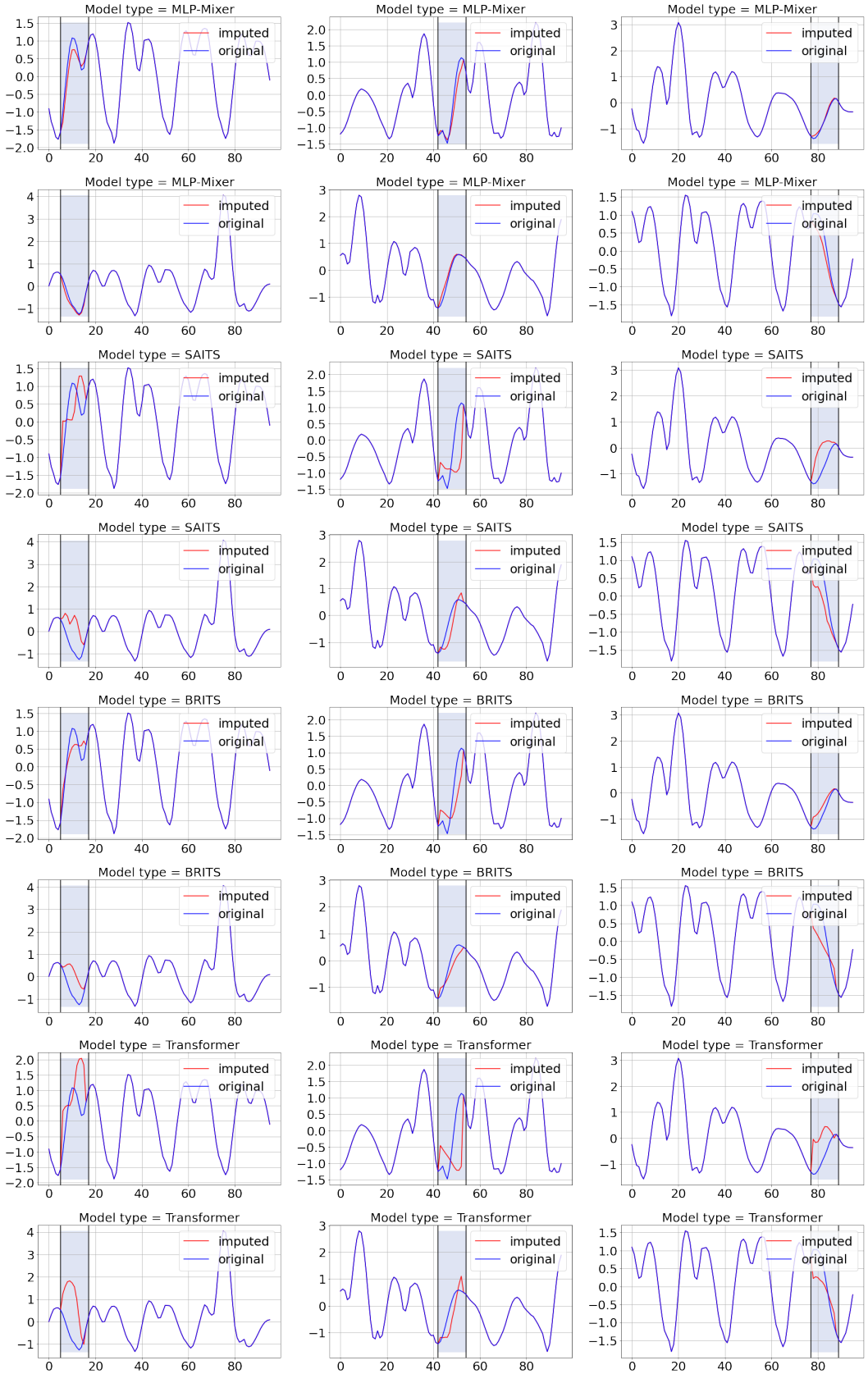


Figure 4: (Traffic) Imputation quality with different imputation models. We set $L_{\text{mask}} = 10\%$. Original and imputed refer to the original time series and the imputed time series respectively.