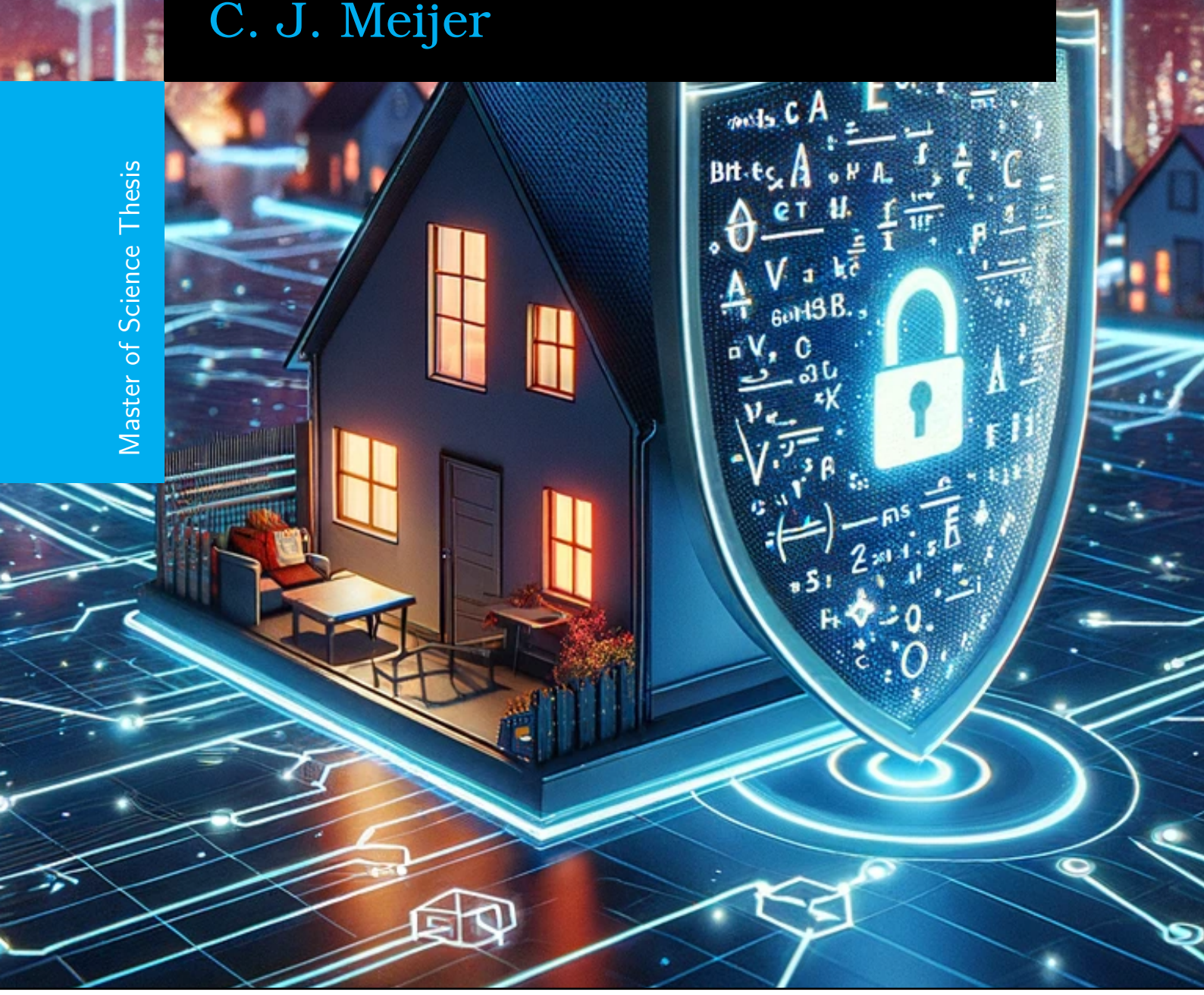
Attacking Federated Time Series Forecasting Models

Reconstructing Private Household Energy Data during Federated Learning with Gradient Inversion Attacks

C. J. Meijer

Master of Science Thesis



TU Delft Delft University of Technology

Distributed Machine Learning Systems

Attacking Federated Time Series Forecasting Models

Reconstructing Private Household Energy Data during Federated Learning with Gradient Inversion Attacks

by

C. J. Meijer

For the degree of Master of Science in Computer Science at Delft University of Technology

to be defended publicly on Wednesday July 10, 2024 at 14:00.

Student number:	4719298	
Project duration:	8 January 2024 - 10 July 2024	
Thesis committee:	Dr. L. Y. Chen,	TU Delft, Chair/Supervisor
	Dr. C. C. S. Liem,	TU Delft, Committee
	Ms. J. Huang,	TU Delft, Daily Co-supervisor

Faculty of Electrical Engineering, Mathematics and Computer Science (EWI)

An electronic version of this thesis is available at https://repository.tudelft.nl/.

The work in this thesis was supported by TNO. Their cooperation is hereby gratefully acknowledged.

Abstract

Federated learning for time series forecasting enables clients with privacy-sensitive time series data to collaboratively learn accurate forecasting models, e.g., in energy load prediction. Unfortunately, privacy risks in federated learning persist, as servers can potentially reconstruct clients' training data through gradient inversion attacks. While gradient inversion attacks are demonstrated for image, text and tabular classification tasks, little is known for time series regression tasks. In this paper, we first conduct an extensive empirical study on inverting time series data across 4 time series forecasting models and 4 datasets, identifying the unique challenges of reconstructing both observations and targets of time series data. We then propose TS-Inverse, a novel gradient inversion attack that improves the inversion of time series data through (i) learning a gradient inversion model that outputs quantile predictions, (ii) a unique loss function incorporating periodicity and trend regularization, and (iii) regularization according to the quantile predictions. Our evaluations demonstrate a remarkable performance of TS-Inverse, achieving at least 2x-10x improvement in terms of sMAPE metric over existing gradient inversion attacks methods on time series data.

Preface

I would like to express my heartfelt gratitude to my academic thesis supervisor, Prof. Lydia Y. Chen, and my daily supervisor, PhD candidate Jiyue Huang, for their invaluable guidance and motivational discussions, which provided me with constant encouragement.

Furthermore, I would like to sincerely thank Shreshtha Sharma and Elena Lazovik for providing me with the opportunity to conduct research at the Advanced Computing Engineering (ACE) Department as part of my internship at TNO. I am also deeply grateful towards my colleague, Paolo Pileggi, for his sincere advice and the engaging discussions, as well as my other colleagues at ACE for their thoughtful questions and stimulating conversations.

On a personal note, I would like to thank my girlfriend, brother, and friends for their encouragement and engagement with my ideas throughout this journey. A special thank you goes to my parents, who have shown me the world and instilled in me the intrinsic motivation to strive for my best self.

Reflecting on the journey, I feel privileged to have had the opportunities that have brought me to this point, and I acknowledge the significant role these experiences have played in my personal and professional growth.

Delft, University of Technology — C. J. Meijer
July 3. 2024

Table of Contents

Chapter 1

Problem Statement

The energy transition has highlighted the need for better grid optimization, particularly at the household level [1, 2]. While high voltage grid optimization is already in practice, improvements are needed for low voltage grids [3]. A key aspect of grid optimization is forecasting energy demand, which can be approached as a time series forecasting problem, using historical data to predict future consumption [4, 5, 6]. A specific use case, researched at TNO, for load forecasts is predicting transformer overloads through the use of a grid simulator, which allows system distribution operators to predict and optimize the low-voltage grid [7].

However, European laws (GDPR) restrict the collection of individual energy consumption data, making it challenging to achieve accurate forecasts on low voltage grids [8, 9]. To address this researchers, in general and at TNO, are exploring methods that allow load forecasting without collecting personal data [3, 10, 11]. One promising solution is federated learning [10].

Federated learning is a training technique where data remains distributed, but a global model is trained on each local dataset [12]. Instead of centrally collecting data to train a single model, federated learning iteratively trains models locally and then collects and aggregates these locally trained models centrally.

Despite its advantages, federated learning still poses challenges and privacy risks [13]. Transferring models instead of data can still reveal information about the local training data [14]. Most research on privacy breaches has been conducted in the domain of images or text [14]. Researchers are actively studying the extent of information leakage from these model updates [14, 15].

In the context of low voltage load forecasting, it is most important to evaluate the possibilities of exactly reconstructing the energy load data from the federated data [3]. In terms of privacy breaches in federated learning, the exact reconstruction of training data can be achieved through gradient inversion attacks [16]. These attacks assume an "honest-but-curious" server that follows the federated learning framework rules but analyzes updates to extract information [16].

This thesis aims to investigate the privacy risks associated with federated time series forecasting, with load forecasting as the primary use case, from an attacker's perspective. The

primary objectives are to evaluate existing gradient inversion attacks on time series data and forecasting models. Additionally, a novel attack, named TS-Inverse, is proposed as a gradient inversion attack tailored specifically for time series forecasting. The forecasting models will be evaluated based on their information leakage through federated model updates.

To achieve these objectives, the research will address the following questions:

- **RQ1: How effective are existing gradient inversion attacks for federated time series forecasting situations?**
- **RQ2: How can gradient inversion attacks be tailored to improve the reconstruction of time series data?**
- **RQ3: How does the architecture of a forecasting model influence its vulnerability to gradient inversion attacks?**

In the subsequent chapters our contributions are presented: In Chapter 2 the main contribution is given in the form of a paper, Chapter Chapter 3 functions as supplementary background information elaborating on topics discussed in the paper. Chapter 4 presents the results of additional experiments conducted during the research, and finally Chapter 5 offers the conclusion of the study.

Chapter 2

Research Paper

This chapter present the paper titled "TS-Inverse: A Gradient Inversion Attack Tailored for Federated Time Series Forecasting Models" which has been submitted to the 24$^{\text{th}}$ International Conference of Data Mining (ICDM 2024) in Abu Dhabi. The paper addresses the privacy risks in federated learning for time series forecasting by proposing TS-Inverse, a novel gradient inversion attack tailored for time series data.

TS-Inverse: A Gradient Inversion Attack tailored for Federated Time Series Forecasting Models

Caspar Meijer*†, Jiyue Huang*, Shreshtha Sharma†, Elena Lazovik†, and Lydia Y. Chen*
*Delft University of Technology, Delft, The Netherlands
†ACE Department, TNO, The Hague/Groningen, The Netherlands

***Abstract*—Federated learning (FL) for time series forecasting (TSF) enables clients with privacy-sensitive time series (TS) data to collaboratively learn accurate forecasting models, e.g., in energy load prediction. Unfortunately, privacy risks in FL persist, as servers can potentially reconstruct clients' training data through gradient inversion attacks (GIA). While GIA is demonstrated for image classification tasks, little is known for time series regression tasks. In this paper, we first conduct an extensive empirical study on inverting TS data across 4 TSF models and 4 datasets, identifying the unique challenges of reconstructing both observations and targets of TS data. We then propose TS-Inverse, a novel GIA that improves the inversion of TS data through (i) learning a gradient inversion model that outputs quantile predictions, (ii) a unique loss function incorporating periodicity and trend regularization, and (iii) regularization according to the quantile predictions. Our evaluations demonstrate a remarkable performance of TS-Inverse, achieving at least 2x-10x improvement in terms of sMAPE metric over existing GIA methods on TS data. Code repository: https://github.com/Capsar/ts-inverse**

***Index Terms*—Federated learning, time series, forecasting, gradient inversion attack**

I. Introduction

Federated learning (FL), a distributed machine learning framework where a server and multiple clients collaboratively train a global model, is established as an effective method for training deep neural networks without centrally storing client training data. In this framework, clients independently train the model on their local data, and then transmit these updates to a central server. The server aggregates these updates and dispatches the new global model back to the clients completing the global training round [1]. This learning paradigm has wide applications in industries that face privacy issues in collecting raw data, such as the energy distribution industry [2].

FL for time series forecasting. The energy industry actively researches data-driven technologies for load forecasting, a typical time series (TS) data challenge, where historical data is utilized to predict future load patterns [2], [3]. Specifically the forecasting of individual loads on low-voltage (smart-)grids is a key subject of interest within the community as it allows the Distribution System Operators to optimize the grid on a local level [3], [4]. Local-level forecasting, while beneficial, faces a legal challenge: The EU general data protection regulation (GDPR) protects the privacy of household and corporate energy data, preventing its central aggregation for ML model training [5], [6]. This is where FL comes into play, allowing the data to remain at its source.

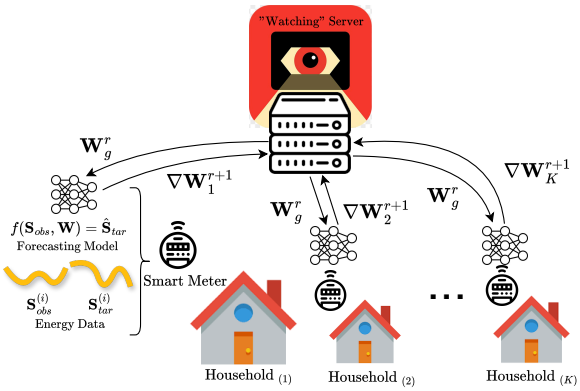


Fig. 1: Federated time series forecasting: a use case in energy demand forecasting with an *honest-but-curious* "watching" server. Each smart meter has a forecasting model and a sequence of energy data. The server distributes the global model parameters and retrieves the locally trained models.

Figure 1 illustrates an example scenario where households are equipped with smart meters that record private energy data, participating as clients in the FL system. The objective of the FL system is to build a global forecasting model that predicts future energy consumption according to the historical observations of energy traces by all clients.

Nonetheless, FL is not without privacy risks, such as membership inference, and property inference attacks [7], [8]. This paper specifically addresses gradient inversion attacks (GIA), where attackers aim to reconstruct private training data from client updates [9] by matching clients' gradients with so-called dummy gradients. Previous research has demonstrated the feasibility of such attacks in image classification tasks, using model parameter updates from FL training to reconstruct training data [10]–[19]. Though such GIAs are also an important topic for regression tasks such as load forecasting, little is known about the risk of inverting TS data.

In this paper, we address this gap by focusing on GIAs for federated time series forecasting (TSF) models. Inverting TS poses unique challenges compared to inverting image data with classification tasks. The training data of TS is composed of a sequence of observations and target values, whereas the existing GIA deals with input pairs of images and labels. Reconstructing TS thus requires inverting gradients

into the sequences of observations and targets. The network architecture for TSF includes components to capture time dependency, e.g., Gated Recurrent Unit (GRU) [20] and 1-D temporal convolution (TCN) [21]. These challenges increase with the training batch size. To demystify the inversion risk for TSF, we start with an empirical analysis by applying existing GIAs on combinations of 4 TSF models and 4 datasets. Based on the insights observed, we propose a novel and effective inversion framework, `TS-Inverse`, which utilizes additional knowledge and TS related characteristics for regularization. We also derive the analytical inversion of target values for the special case with a batch size of one. Extensive evaluation results against existing GIAs and ablation studies show that `TS-Inverse` is able to reconstruct both the observation and target values of time series, achieving low reconstructing errors.

Our contributions can be summarized as follows:

- We conduct the first empirical analysis of GIAs on federated TSF models, highlighting the challenges of inverting both observation and target sequences with respect to model architecture, gradient distance, and regularization. (Section III)
- We design `TS-Inverse`, which includes two innovative components: the gradient inversion model and TS-regularized inversion optimization. The gradient inversion model provides relevant quantile bounds by leveraging auxiliary data. The gradient distance is the L1-Norm with TS regularization incorporating periodicity, trend, and the learned quantile bounds of the gradient inversion model. Additionally, a one-shot analytical technique is used for reconstructing targets. (Section IV)
- We extensively evaluate `TS-Inverse` on four datasets and five forecasting architectures, demonstrating a 2x-10x reduction in reconstruction error on the non-RNN-based architectures. (Section V)

II. Related Studies

A. Federated Learning for Time series Forecasting

Most of the existing FL studies focus on image/text classification tasks [8], with only a few addressing TS data [22], [23], which mainly concerns the tasks of forecasting. Specifically, TSF tasks predict future values based on historically time-stamped data of continuous TS, e.g., Energy load forecasting [22], [23]. Model architectures to study TSF include standard statistical models, such as ARIMA [24], as well as deep learning methods. Among the deep learning methods, RNNs [25], including GRU [20] and Long Short-Term Memory (LSTM) [26] are commonly used on TSF task, [22], [27]. while recent studies also indicate the effectiveness of non-recurrent models like TCN [21], [28].

Inherited from the nature of FL, learning TSF also involves the non-identical and independently distributed client data [22], [23], [29]. Taking energy load forecasting as an example, each client, such as households and company buildings, possesses a distinct load profile. Each client only contains its own unique load data, different from cases where multiple customers can reside in a single client data silo. Although the non-iidness of load profiles and their accompanying convergence issues are beyond the scope of this paper, it is crucial to note that each client's load profile is private and must be protected against privacy leakage [30], [31]. Notably, these works considering the privacy of load forecasting in FL merely discusses differential privacy budgets and accuracy differences, without evaluating any actual attacks [30], [31].

B. Privacy and Inversion Attacks on Federated Learning

Compared with adversarial attacks that aim to modify and disrupt the learning tasks [8], privacy attacks in FL try to infer private information from the system [8], e.g., membership inference attacks [32] and property inference attacks [33]. Among these privacy attacks, data reconstruction attacks [10]–[19] that aim to recover training samples from clients are most harmful and is our focus of this paper. Such reconstruction attacks leverage gradients collected from clients, hence known as gradient inversion attacks. Besides, there are no existing studies that address these inversion risks of TS data.

Existing work on inversion attacks can be categorized into two types: pure optimization-based and informed optimization-based as presented in Table I. Both categories optimize dummy data such that their gradients match the

Approach	Attack Name / Source	Type of attack	Space	Gradient Distance	Regularization	FL Training Loss
Pure	DLG [10]	Optimization	$(\mathbf{x}, \mathbf{y})$	L2	-	Cross Entropy
	SAPAG [11]	Optimization	$(\mathbf{x}, \mathbf{y})$	Weighted Gaussian Kernel	-	Cross Entropy
	iDLG [12]	Optimization	$(\mathbf{x})$	L2	-	Cross Entropy
	Inverting Gradients [13]	Optimization	$(\mathbf{x})$	Cosine	TV	Cross Entropy
	TAG [14]	Optimization	$(\mathbf{x}, \mathbf{y})$	L2 + α L1	-	Causal (Cross Ent.)
	General DL [17]	Optimization	$(\mathbf{x})$	L2	TV, Clip, Scale	Cross Entropy
	DIA [18]	Optimization	$(\mathbf{x})$	Cosine	TV, Mask	Cross Entropy
Informed	GradInversion [15]	(Latent) Optimization	$(\mathbf{z}, \mathbf{x})$	L2	TV, BN & Group	Cross Entropy
	GIAS / GIML [16]	(Latent) Optimization	$(\mathbf{z}, G_\theta)$	Cosine	TV, L2	Cross Entropy
	LTI [19]	Learning	$(\mathbf{x})$	Permutation Invariant CE or L2	-	-
	TS-Inverse (this work)	Learning & Optimization	$(\mathbf{x}, \mathbf{y})$	L1	Periodicity, Trend & QB	MSE

TABLE I: Comparison of related gradient inversion attacks and `TS-Inverse`. The space, indicates whether the attack optimizes the inputs ($\mathbf{x}$), outputs ($\mathbf{y}$), latent ($\mathbf{z}$), or generative model parameters (G_θ). The regularization terms mentioned are Total Variation, Clipping, Scaling, Dropout Masks, Batch and Group Normalization, Periodicity, Trend and Quantile Bounds. The FL Training Loss indicates with which loss the global model is trained.

observed gradients. The informed optimization-based attacks further leverage external knowledge (e.g., generative models) to assist the reconstruction process.

The pioneering work of pure optimization-based method, Deep Leakage from Gradients (DLG) [10], formalizes the problem by minimizing the L2-Norm between the original and dummy gradients by optimizing both inputs and labels. Further improvements over DLG involve modifying the loss function, adding regularization terms, or increasing the information used in the attack through priors or by gathering multiple gradients. Wang *et al.* [11] tackle convergence issues in DLG by using a weighted Gaussian kernel function, for normally initialized model parameters. Alongside, iDLG [12] improves reconstruction by analytically inferring sample labels, although it is limited to a batch size of one. Similarly, Geng *et al.* [17] addresses label recovery in realistic scenarios where the batch size is greater than one and duplicate labels are possible. With Inverting Gradients (InvG), Geiping *et al.* [13] introduces the cosine similarity loss and a regularization term to minimize total variation in neighboring pixels, enhancing reconstruction for image data. For NLP models, [14] develops gradient leakage attacks targeting transformer-based models by optimizing embeddings and adding an $L1$ norm to the loss function. Finally, Scheliga *et al.* [18] expand the optimization space to include dropout masks, leading to the development of the Dropout Inversion Attack (DIA), which enhances attacks on networks with dropout layers.

Further advancements using additional models include GradInv [15], which uses batch classification label restoration with fidelity and group consistency regularization and it relies on a pre-trained generative model. [16] also leverages a pre-trained generative model, but goes further in showing it can train such a model from gradients only. Learning to Invert (LTI) [19] trains an inverting model that maps the gradient to the training data according to an auxiliary dataset, focusing on countering defenses.

Despite these advancements, which focus on image classification or text processing tasks, it remains unsure whether TS data can be recovered according to the TS gradients. Additionally, no current research addresses the inversion of architectures specifically for TS data, such as TCNs or GRUs.

III. Empirical Inversion Analysis on Time series Forecasting

In this section, we introduce and analyze the risk of inverting clients' private TS data in the context of federated forecasting by applying existing classification-based attacks. We first define TSF and outline the assumptions made regarding the FL setups and threat model. Afterward, we formally define the gradient inversion attacks under our forecasting scenarios. Finally, we analyze the feasibility and risks of existing image-based inversion attacks on different time series models.

A. Problem Definition

1) Time-series Forecasting: TS data are generally represented through discrete time steps, determined by a chosen sampling frequency, though time is inherently continuous. TSF is a regression task that predicts future values given a historical segment. In this work, we focus on multi-step TSF where the future segment consists of multiple timesteps. Formally, the historical segment, comprising H steps, is defined as a series of subsequent observation timesteps $obs = \{t \in \mathbb{Z} \mid -H < t \leq 0\}$, whereas the future segment, spanning F steps, is denoted as the targets $tar = \{t \in \mathbb{Z} \mid 0 < t \leq F\}$ as illustrated in Figure 2. We represent historical data of the sequence $\mathbf{S}$ as $\mathbf{S}_{obs} \in \mathbb{R}^{H \times d}$, and the forecasting target data as $\mathbf{S}_{tar} \in \mathbb{R}^{F \times d}$, where d indicates the number of distinct features. A time series is univariate when $d = 1$, and multivariate when $d > 1$. Furthermore, $\mathbf{S}_t \in \mathbb{R}^d$ is used to denote a value vector at timestep t. We focus on the univariate case. For training the TSF task, we consider recent advanced deep learning models. The training objective of those models is to minimize the mean square errors (MSE) or mean absolute errors (MAE).

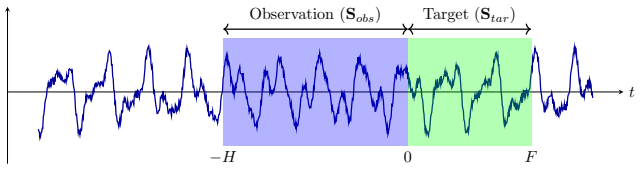


Fig. 2: Example of univariate time-series forecasting.

2) FL for Time-series Forecasting: A federated forecasting framework involves K clients, each having its own data silo with sequence data $\mathbf{S}_k$. Each client's sequence $\mathbf{S}_k$ is split up in multiple subsequent pairs of observation data $\mathbf{S}_{obs}^{(i)}$ and target data $\mathbf{S}_{tar}^{(i)}$, where i denotes the i'th sample from the sequence. How these samples are gathered is explained in the experimental setup in section V-A2. A client's data silo contains a dataset with sequence samples organized as $\{\mathbf{S}_{obs}^{(i)}, \mathbf{S}_{tar}^{(i)}\}_{i=1}^{n_k}$, where n_k denotes the number of local sequence pairs. The clients jointly train a global forecasting model via a strategy determined by the server, e.g., learning rate and batch sizes.

The server initializes a global model with parameters $\mathbf{W}_g^r$, where g stands for the global model and r denotes the global training round. The server distributes $\mathbf{W}_g^r$ to the clients, who then follow federated stochastic gradient decent (FedSGD) [1] to further train the model using their own data. Specifically, the clients perform a single gradient update on a single batch, resulting in gradients $\nabla \mathbf{W}_k^r$. These are sent back to the server, which in turn, aggregates all model updates as $\mathbf{W}_g^{r+1} = \mathbf{W}_g^r - \alpha \left(\frac{1}{K} \sum_{k=1}^{K} \nabla \mathbf{W}_k^r \right)$, where α is the global learning rate.

B. Threat model

In line with the related studies, the threat model for our gradient inversion on TSF models assumes an *honest-but-curious* server. This means that while the server follows protocol rules, it is interested in extracting training data client information from the gradients data it receives. The attack considered is a *white-box* attack, where the FL strategy, model architecture, optimizer, loss function, observations, and target

size ($\mathbb{R}^{d\times H}, \mathbb{R}^{d\times F}$), learning rate ($\alpha$), batch size ($\mathcal{B}$), etc. are all known to the adversary. The adversary has access to the model weights ($\mathbf{W}_g^r$) and aggregated gradients ($\nabla\mathbf{W}_k^r$) at communication time and aims to reconstruct each sample from the batched data, including both observations and targets ($\mathbf{S}_{obs}, \mathbf{S}_{tar}$). We also assume that the adversary has access to an auxiliary dataset ($\mathcal{D}_{aux}$) with a distribution overlapping with the client's data. Furthermore, the adversary has the computational capabilities to train models outside the FL framework.

C. Attacking Time-series Forecasting

Inverting training data for TSF presents several new challenges compared to inverting training data for classification tasks. The training objectives and inputs differ: for image classification, the objective is cross-entropy loss with pairs of images and their class labels, whereas for time series forecasting, it is mean squared error (MSE) loss with pairs of observation and target series. In image classification, due to the cross-entropy loss, the labels can be analytically inferred from the gradients, as demonstrated by Zhao *et al.* [12] and Geng *et al.* [17]. Consequently, the inversion attack on a classification model primarily involves reconstructing the training images. However, inverting the training data for time series forecasting requires reconstructing both the input observations and the target values ($\mathbf{S}_{obs}$ and $\mathbf{S}_{tar}$).

To recover the observation and target series data, the adversary first initializes a dummy input ($\widetilde{\mathbf{S}}_{obs}$) and target ($\widetilde{\mathbf{S}}_{tar}$) pair with the same dimensions as $\mathbf{S}_{obs}$ and $\mathbf{S}_{tar}$, respectively, and calculates the corresponding gradients:

$$\nabla\widetilde{\mathbf{W}} = \frac{\partial\mathcal{L}(f(\widetilde{\mathbf{S}}_{obs}, \mathbf{W}), \widetilde{\mathbf{S}}_{tar})}{\partial\mathbf{W}}, \tag{1}$$

where f is the global model and $\mathbf{W}$ are its parameters before applying the gradient updates. f is a twice differentiable neural network and $\mathcal{L}$ is the regression loss function used in optimizing f. These dummy observations and targets are optimized by minimizing a distance D between the dummy gradients $\nabla\widetilde{\mathbf{W}}$ from (1) and clients' gradients $\nabla\mathbf{W}$:

$$(\widetilde{\mathbf{S}}^{\star}_{obs}, \widetilde{\mathbf{S}}^{\star}_{tar}) = \underset{\widetilde{\mathbf{S}}_{obs}, \widetilde{\mathbf{S}}_{tar}}{\arg\min}\, D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}). \tag{2}$$

In this work, we define the optimization problem for recovering both the observations ($\mathbf{S}_{obs}$) and the targets ($\mathbf{S}_{tar}$). Specifically, we use the gradient to recover data for both observations and targets. Additionally, the TS data is often pre-processed to be within the domain [0, 1], which helps to bound the reconstructed data. DLG [10] and others [12], [14], [15], [17] uses L2-Norm to match the gradients as:

$$D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}) = \left\| \nabla\widetilde{\mathbf{W}} - \nabla\mathbf{W} \right\|_2. \tag{3}$$

Inverting Gradients [13] and others [16], [18] use the Cosine Similarity loss together with the image prior regularization, total variation, $\mathcal{R}_{\text{TV}}$ with hyperparameter λ_{TV}:

$$\mathcal{D}_{\text{cosine}}(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}) = 1 - \frac{\nabla\widetilde{\mathbf{W}} \cdot \nabla\mathbf{W}}{\|\nabla\widetilde{\mathbf{W}}\|_2 \|\nabla\mathbf{W}\|_2} + \lambda_{\text{TV}}\mathcal{R}_{\text{TV}}. \tag{4}$$

D. Empirical Findings

We present our empirical findings from applying four existing attacks— (designed for classifiers) DLG [10], InvG [13], DIA [18], and LTI [19]— on three models: CNN, FCN, and TCN for inverting the TS data from FL gradients. These attacks differ in their methodologies and gradient distance functions. Specifically, we discuss the impact of model architecture, loss function, and total variation for inversion performance. The experimental setups are detailed in section V-A. We use the Symmetrical Mean Absolute Percentage Error (sMAPE↓) as the evaluation metric to compare all baselines, presented in Table III. Specifically, to demonstrate the impact of the total variation regularization ($\mathcal{R}_{\text{TV}}$) for InvG, we report the sMAPE results in Table II. Additionally, we also visualize the recovered observation and target data for easy perceptual comparison, by a single TS example in Figure 3. In-depth comparison is summarized in Table III and analyzed in Section V.

Model Architectures The TCN architecture [21] includes dropout layers, which make the optimization attack more difficult, according to [34]. DIA [18] counters this dropout defense by also optimizing the dropout masks that indicate which neurons are dropped during the training round. This is required in order to accurately reconstruct training data from architectures with dropout layers. Moreover, GRUs are designed to selectively remember and forget information, a process controlled through nonlinear functions governed by reset and update gates [20]. These gates determine the flow and modification of information over time, inherently leading to a loss of information and many-to-one mappings from input sequences to hidden states. From a gradient perspective, while gradients provide insights into how changes in inputs could

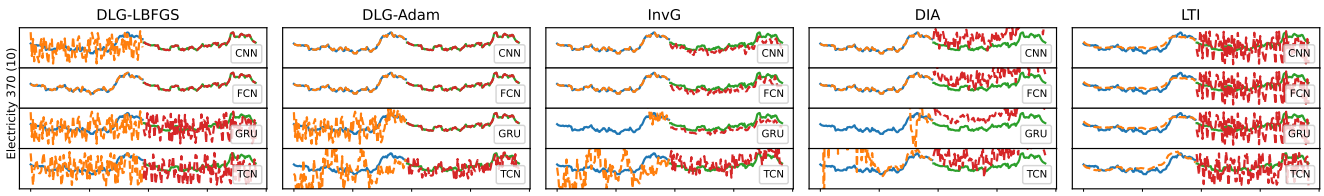


Fig. 3: Baseline reconstruction results with FCN, CNN, GRU-2-FCN, and TCN model architectures on the Electricity 370 dataset with seeds (10). In each graph, the blue and green lines are the ground truth observations and targets respectively. The orange and red lines are the respective reconstructed observations and targets.

affect the outputs, they only do so on average over the whole input space and do not offer direct mappings to individual inputs. This many-to-one relationship also applies to RNN architectures.

Gradient Distance Function of Attacks The effects of the gradient distance function are most clear for the CNN and FCN architectures, according to Table III and Figure 3. The DLG attack, which uses L2-Norm to align gradients, effectively reconstructs the observations and targets. This is because L2-Norm measures the squared differences between gradients, inherently capturing both the magnitude and the direction of the gradients. In contrast, the InvG and DIA attacks rely on Cosine Similarity, which normalizes the gradients and focuses only on the direction, disregarding their magnitude. This focus on direction alone results in noisier targets, indicating that the magnitude of gradients is essential for accurate target reconstruction.

Total Variation The application of total variation, as proposed by [13], is observed to have a counterproductive effect on the reconstruction of TS data, as shown in Table II. The regularization results in a higher sMAPE value when applied to TS. We applied the total variation separately to the observations and targets using hyperparameters $\lambda_{\text{TV}}^{obs}$ and $\lambda_{\text{TV}}^{tar}$, respectively. When increasing $\lambda_{\text{TV}}^{obs}$ while keeping $\lambda_{\text{TV}}^{tar} = 0$, the sMAPE increased for the observations, while for the targets it remained the same. In contrast, when increasing $\lambda_{\text{TV}}^{tar}$ while keeping $\lambda_{\text{TV}}^{obs} = 0$, the sMAPE for observations increased, while it decreased for targets. The total variation reduces noise in images, which helps improve the image reconstruction quality. However, this does not translate well to TS data. This is because TS data can be inherently noisier than images due to large differences between subsequent timesteps.

Highlights

1. TCN and GRU architectures are more challenging to invert compared to CNN and FCN due to their inherent design features like dropout layers and gate mechanisms. **2.** To correctly recreate targets, the gradient distance has to take into account the magnitudes of the gradient values. **3.** Image regularization priors are not effective for time series data, which tends to be intrinsically noisier and exhibit larger differences between subsequent timesteps.

IV. TS-Inverse Framework

Following the findings from the empirical study, we propose a first-of-kind gradient inversion attack for federated TSF, named TS-Inverse, which inverts a batch of observation and target data of time series regression tasks from clients gradients. We illustrate the framework of TS-Inverse in Figure 4, consisting of three key steps. The first component of TS-Inverse is to train a gradient inversion model, i.e., f_{inv}, using auxiliary data to map model gradients to sequences of values that correspond to specific quantile ranges of the observations and targets, as depicted in Figure 7. The quantile

	Dataset	Electricity 370		Proprietary Dataset	
$\lambda_{\text{TV}}^{obs}$	$\lambda_{\text{TV}}^{tar}$	Observation	Target	Observation	Target
0	0	$\mathbf{0.002}_{0.00}$	$0.094_{0.06}$	$\mathbf{0.016}_{0.00}$	$1.143_{0.23}$
	0.001	$0.100_{0.01}$	$0.042_{0.04}$	$0.733_{0.16}$	$1.040_{0.25}$
	0.01	$0.797_{0.04}$	$\mathbf{0.009}_{0.00}$	$1.528_{0.07}$	$0.036_{0.02}$
0.001	0	$0.029_{0.01}$	$0.094_{0.06}$	$0.170_{0.02}$	$1.144_{0.23}$
	0.001	$0.067_{0.01}$	$0.042_{0.04}$	$0.467_{0.14}$	$1.042_{0.25}$
	0.01	$0.501_{0.06}$	$\mathbf{0.009}_{0.00}$	$1.394_{0.06}$	$\mathbf{0.028}_{0.01}$
0.01	0	$0.029_{0.00}$	$0.094_{0.06}$	$0.165_{0.02}$	$1.145_{0.23}$
	0.001	$0.067_{0.00}$	$0.042_{0.04}$	$0.472_{0.13}$	$1.042_{0.25}$
	0.01	$0.500_{0.06}$	$\mathbf{0.009}_{0.00}$	$1.398_{0.06}$	$\mathbf{0.028}_{0.01}$

TABLE II: Comparison of the effects of total variation using the InvG attack measured with sMAPE ↓ with CNN architecture and Batch size $\mathcal{B} = 1$ and seeds (10, 43, and 28).

ranges divide the data into intervals with equal probabilities, giving a representation of the observation and target distribution that belongs to the gradients. These sequences serve as upper and lower bounds for the learned quantile regularization. The second key feature is the inversion optimization, starting from the dummy time series. Different from the convention of minimizing L2-Norm and cosine distance in image inversion attacks, we advocate using L1-Norm loss and combine it with the proposed time series specific regularization terms, namely periodicity, trend, and quantile bounds. Furthermore, we provide an analytical one-shot inversion technique for the target sequence, under the scenario where the batch size is 1.

A. Gradient Inversion Model

We assume the existence of an auxiliary dataset $\mathcal{D}_{aux}$ by the *honest-but-curious* server. This is ideally a subset of the time series of the data silos that are being attacked, but it can also be an out-of-distribution dataset [19]. The dataset is first pre-processed such that the samples have the same sequence length as the client's data samples through interpolation.

We define a gradient inversion model $f_{inv} : \mathbb{R}^m \rightarrow (\mathbb{R}^{H\times d\times Q}, \mathbb{R}^{F\times d\times Q})$, where m is the number of parameters in the forecasting model f, and Q is the number of quantile bounds. This model is designed to map gradients to quantile bounds, which are sequences of values corresponding to specific quantile levels of the TS observations and targets. Quantile levels (τ) are probability values between 0 and 1. For example, between the bounds corresponding to quantile levels 0.1 and 0.9 fall 80% of the sequences.

The model f_{inv} is trained on the gradients of the forecasting model f derived from batches of samples from $\mathcal{D}_{aux}$. A batch $(\mathbf{S}_{obs}^{\mathcal{B}}, \mathbf{S}_{tar}^{\mathcal{B}})$ consists of observation and target pairs from $\mathcal{D}_{aux}$, where $\mathcal{B}$ denotes the number of such pairs.

The model f_{inv} is optimized using the pinball loss with respect to the batched samples. It produces the outputs $\{\hat{\mathbf{S}}_{obs}^{\tau_1}, \hat{\mathbf{S}}_{obs}^{\tau_2}, \ldots, \hat{\mathbf{S}}_{obs}^{\tau_Q}\}$ and $\{\hat{\mathbf{S}}_{tar}^{\tau_1}, \hat{\mathbf{S}}_{tar}^{\tau_2}, \ldots, \hat{\mathbf{S}}_{tar}^{\tau_Q}\}$, where each τ_q represents a specific quantile level. These quantile predictions are repeated along the batch dimension to match $(\mathbf{S}_{obs}^{\mathcal{B}}, \mathbf{S}_{tar}^{\mathcal{B}})$. As a result the quantile predictions demonstrate the variability of the batch. The pinball loss, which is used for

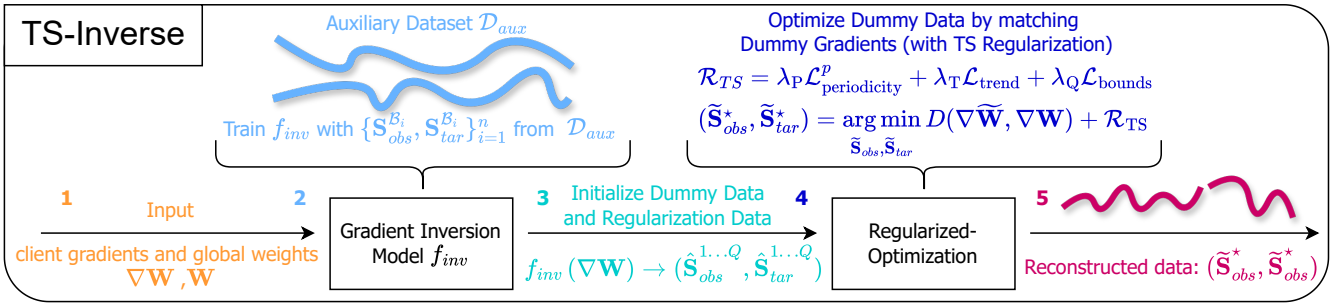


Fig. 4: Key steps of `TS-Inverse` framework

optimization, is defined as follows:

$$\mathcal{L}^{\tau}_{\text{pinball}}(\mathbf{S}, \hat{\mathbf{S}}) = \frac{1}{T}\sum_{t=1}^{T} \max\Big((\tau-1)(\mathbf{S}_t - \hat{\mathbf{S}}_t), \tau(\mathbf{S}_t - \hat{\mathbf{S}}_t)\Big), \tag{5}$$

where t is a timestep and T is the total number of timesteps in sequence $\mathbf{S}$, and $\hat{\mathbf{S}}$ represents the predicted quantile bound. The total loss is to apply (5) for each quantile level over the observations and targets as:

$$\mathcal{L}_{inv} = \sum_{q=1}^{Q} \frac{\mathcal{L}^{\tau_q}_{\text{pinball}}(\mathbf{S}^{\mathcal{B}}_{obs}, \hat{\mathbf{S}}^{\tau_q}_{obs}) + \mathcal{L}^{\tau_q}_{\text{pinball}}(\mathbf{S}^{\mathcal{B}}_{tar}, \hat{\mathbf{S}}^{\tau_q}_{tar})}{2}. \tag{6}$$

By using the pinball loss function, the model f_{inv} is effectively trained to provide quantile bounds for the entire batch of TS data.

B. Gradient Matching Distance Function

The cosine similarity loss only matches the hyperplane direction of the gradients, but does not take into account the magnitudes of the values. Matching the magnitude of the values is essential for gradient approximation. Otherwise, the optimization will be dominated by the large values, resulting in distortion of updating dummy data. One can add an additional term to the cosine similarity loss that takes into account the norm of the difference of the gradients [35], [36].

On the other hand, the L1 loss, i.e., the summed absolute errors, is a single-term metric that is robust to outliers, which is common in gradients. Additionally. it matches the magnitude of the gradient values and it is particularly compatible with models that utilize weights initialized from a normal distribution [14], such as the TCN architecture. Our choice is backed by empirical findings that show that using the L1-Norm leads to better reconstructions compared to the Cosine with or without the L2-Norm. The distance function is thus formulated as follows:

$$D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}) = \left\|\nabla\widetilde{\mathbf{W}} - \nabla\mathbf{W}\right\|_1. \tag{7}$$

	Dataset	Electricity 370		KDDCup		London Smartmeter		Proprietary Dataset	
Model	Attack Method	Observation	Target	Observation	Target	Observation	Target	Observation	Target
CNN	DLG-LBFGS	$0.632_{0.05}$	$0.001_{0.00}$	$1.454_{0.11}$	$0.030_{0.03}$	$1.429_{0.14}$	$0.023_{0.01}$	$1.294_{0.08}$	$0.011_{0.00}$
	DLG-Adam	$0.029_{0.03}$	8.6e-05$_{0.00}$	$1.349_{0.23}$	$0.029_{0.03}$	$0.993_{0.14}$	$0.006_{0.00}$	$1.013_{0.27}$	$0.005_{0.00}$
	InvG	$0.002_{0.00}$	$0.189_{0.13}$	$0.038_{0.02}$	$1.306_{0.30}$	$0.027_{0.00}$	$1.136_{0.26}$	$0.015_{0.00}$	$1.160_{0.19}$
	DIA	$0.002_{0.00}$	$0.542_{0.11}$	$0.073_{0.04}$	$1.293_{0.11}$	$0.094_{0.04}$	$1.501_{0.05}$	$0.018_{0.01}$	$1.484_{0.08}$
	LTI	$0.140_{0.04}$	$0.685_{0.07}$	$0.889_{0.15}$	$1.399_{0.28}$	$0.713_{0.18}$	$1.469_{0.06}$	$0.486_{0.05}$	$1.305_{0.09}$
	TS-Inverse	**0.002**$_{0.00}$	1.2e-05$_{0.00}$	**0.017**$_{0.01}$	2.1e-04$_{0.00}$	**8.1e-05**$_{0.00}$	2.4e-05$_{0.00}$	**0.009**$_{0.00}$	7.1e-05$_{0.00}$
	TS-Inverse$_{\text{one-shot}}$	$0.003_{0.00}$	**1.4e-07**$_{0.00}$	$0.085_{0.05}$	**1.9e-06**$_{0.00}$	$0.024_{0.01}$	**2.1e-06**$_{0.00}$	$0.032_{0.01}$	**7.3e-07**$_{0.00}$
FCN	DLG-LBFGS	$0.025_{0.01}$	2.0e-04$_{0.00}$	$0.168_{0.05}$	$0.002_{0.00}$	$0.498_{0.17}$	$0.003_{0.00}$	$0.181_{0.07}$	$0.002_{0.00}$
	DLG-Adam	2.8e-06$_{0.00}$	1.2e-05$_{0.00}$	$0.002_{0.00}$	4.1e-04$_{0.00}$	$0.004_{0.00}$	2.9e-05$_{0.00}$	1.8e-05$_{0.00}$	3.5e-05$_{0.00}$
	InvG	4.7e-06$_{0.00}$	$0.214_{0.15}$	6.1e-05$_{0.00}$	$1.261_{0.39}$	7.8e-05$_{0.00}$	$1.300_{0.10}$	2.6e-05$_{0.00}$	$1.108_{0.15}$
	DIA	$0.004_{0.00}$	$0.542_{0.10}$	$0.027_{0.02}$	$1.343_{0.19}$	$0.068_{0.05}$	$1.571_{0.07}$	1.1e-04$_{0.00}$	$1.551_{0.07}$
	LTI	$0.133_{0.03}$	$0.685_{0.07}$	$0.604_{0.17}$	$1.399_{0.28}$	$0.673_{0.07}$	$1.469_{0.06}$	$0.498_{0.03}$	$1.305_{0.09}$
	TS-Inverse	**1.7e-06**$_{0.00}$	8.4e-07$_{0.00}$	**2.4e-06**$_{0.00}$	**1.1e-06**$_{0.00}$	**6.3e-06**$_{0.00}$	2.5e-06$_{0.00}$	**6.4e-06**$_{0.00}$	4.4e-06$_{0.00}$
	TS-Inverse$_{\text{one-shot}}$	**3.1e-07**$_{0.00}$	**8.3e-08**$_{0.00}$	5.3e-06$_{0.00}$	1.5e-06$_{0.00}$	3.2e-05$_{0.00}$	**1.4e-06**$_{0.00}$	6.6e-06$_{0.00}$	**8.1e-07**$_{0.00}$
TCN	DLG-LBFGS	$0.652_{0.06}$	$0.689_{0.04}$	$1.483_{0.06}$	$1.374_{0.28}$	$1.437_{0.10}$	$1.429_{0.09}$	$1.321_{0.13}$	$1.243_{0.09}$
	DLG-Adam	$0.726_{0.07}$	$0.362_{0.11}$	$1.320_{0.13}$	$1.072_{0.41}$	$1.136_{0.18}$	$1.079_{0.19}$	$1.141_{0.16}$	$1.028_{0.09}$
	InvG	$0.704_{0.05}$	$0.553_{0.20}$	$1.359_{0.16}$	$1.445_{0.19}$	$1.154_{0.18}$	$1.493_{0.11}$	$1.114_{0.12}$	$1.424_{0.11}$
	DIA	$1.088_{0.31}$	$0.719_{0.20}$	$0.728_{0.20}$	$1.229_{0.07}$	$1.076_{0.57}$	$1.576_{0.01}$	$1.282_{0.34}$	$1.473_{0.05}$
	LTI	$0.158_{0.05}$	$0.628_{0.06}$	$0.839_{0.13}$	$1.437_{0.30}$	$0.743_{0.08}$	$1.402_{0.06}$	$0.530_{0.13}$	$1.326_{0.09}$
	TS-Inverse	$0.097_{0.04}$	$0.007_{0.01}$	**0.509**$_{0.13}$	$0.042_{0.05}$	$0.194_{0.22}$	$0.106_{0.19}$	**0.172**$_{0.05}$	$0.021_{0.02}$
	TS-Inverse$_{\text{one-shot}}$	**0.089**$_{0.03}$	**1.2e-07**$_{0.00}$	$0.559_{0.21}$	**1.3e-06**$_{0.00}$	**0.188**$_{0.12}$	**1.8e-06**$_{0.00}$	$0.251_{0.14}$	**7.6e-07**$_{0.00}$

TABLE III: Comparison of sMAPE ↓ with baseline attacks and TS-Inverse on multiple datasets and model architectures with each 5000 attack steps. Batch size $\mathcal{B} = 1$. The seeds used are: (10, 43, 28, 80 & 71)

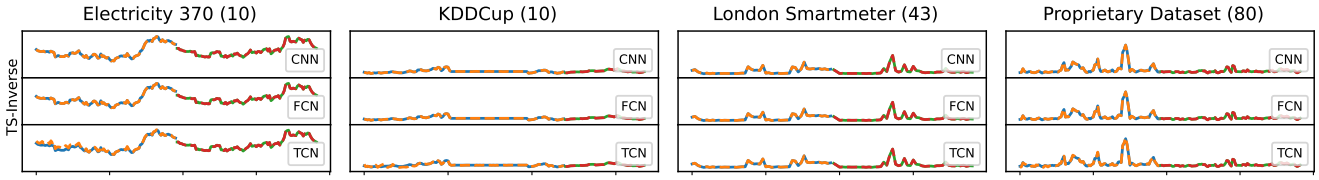


Fig. 5: `TS-Inverse` reconstruction results with FCN, CNN, and TCN model architectures on a sample from each dataset.

C. Regularizations: Periodicity, Trend and Quantile Bounds

In time series, there are data-specific characteristics that can be leveraged to regularize the reconstruction: periodicity and trend. The former refers to the repeating patterns or cycles in the data over a specific period, whereas the trend represents the long-term progression or direction in the data. Furthermore, the learned quantile ranges, outputted by f_{inv}, are used as soft bounds for the reconstructed data. We weight and combine those patterns into one regularization term, $\mathcal{R}_{TS}$:

$$\mathcal{R}_{TS} = \lambda_{\text{P}}\mathcal{L}^{p}_{\text{periodicity}} + \lambda_{\text{T}}\mathcal{L}_{\text{trend}} + \lambda^{obs}_{\text{Q}}\mathcal{L}^{obs}_{\text{bounds}} + \lambda^{tar}_{\text{Q}}\mathcal{L}^{tar}_{\text{bounds}}, \tag{8}$$

where $\lambda_{\text{periodicity}}$, λ_{trend}, λ^{obs}_{Q}, and λ^{tar}_{Q} are hyperparameters controlling the importance of periodicity, trend and learned quantile bounds regularizations, respectively.

1) Periodicity Regularization: Understanding the specific periodic nature of the data is crucial and the periodicity must be present in the combined observation and target sequence. To capture periodicity, we minimize the absolute error between each point and its corresponding point one period p apart. Here, p represents the number of timesteps for one full period or cycle. This regularization ensures that reconstructed data maintains the inherent cyclical patterns. Periodicity can vary widely depending on the dataset, exhibiting daily, weekly, or even yearly patterns. The regularization is defined as the mean absolute error between the periodic points:

$$\mathcal{L}^{p}_{\text{periodicity}}(\mathbf{S}) = \frac{1}{T-p}\sum_{t=1}^{T-p}\left|\mathbf{S}_t - \mathbf{S}_{t+p}\right|, \tag{9}$$

where p denotes the period length and T is the total number of timesteps in the sequence $\mathbf{S}$.

2) Trend Regularization: To enforce trend consistency, we fit a linear trend to the sequence and minimize the deviation of the actual sequence from this trend. This approach ensures that the trend is regularized over both the observations and targets combined, capturing the underlying trend dynamics in the data. The regularization is defined as the MAE between the sequence and the trend of the sequence, as such:

$$\mathcal{L}_{\text{trend}}(\mathbf{S}) = \left\|\mathbf{S} - \mathbf{S}_{\text{trend}}\right\|_1, \tag{10}$$

where the $\mathbf{S}_{\text{trend}}$ is calculated using linear regression over $\mathbf{S}$ as such: $\mathbf{S}_{\text{trend}} = \beta\,(s - \bar{s}) + \bar{\mathbf{S}}$. Here, s represents the time indices of the sequence, $\bar{s}$ is the mean of these time indices, $\bar{\mathbf{S}}$ is the mean of the sequence $\mathbf{S}$, and the slope β of the linear trend is given by: $\beta = \frac{\sum(s-\bar{s})\cdot(\mathbf{S}-\bar{\mathbf{S}})}{\sum(s-\bar{s})^2}$.

3) Quantile Bounds Regularization: The predictions made by the gradient inversion model are designed to function as soft bounds for the dummy data as illustrated in Figure 7. The bounds are defined as quantile pairs that are symmetrically spaced around the median. Within these bounds, no regularization is applied. However, if the dummy data falls outside this range, an absolute error is calculated for regularization. This approach intuitively ensures that the dummy data is encouraged to stay within realistic and statistically consistent bounds.

For a sequence $\mathbf{S}$ and bounds $\mathbf{S}_{\text{lower}}$ and $\mathbf{S}_{\text{upper}}$, the regularization loss is defined as:

$$\mathcal{L}_{\text{bounds}} = \sum_{q=1}^{Q}\left\|\max(0, \mathbf{S} - \mathbf{S}_{\text{upper}}) + \max(0, \mathbf{S}_{\text{lower}} - \mathbf{S})\right\|_1, \tag{11}$$

where q indexes over the quantiles with $\mathbf{S}_{\text{lower}} = \mathbf{S}^{\tau_q}$ and $\mathbf{S}_{\text{upper}} = \mathbf{S}^{\tau_{(Q-q)}}$, respectively.

Putting Eq. (9) (10), and (11) into (8) and combining it with (7), `TS-Inverse` employs the following loss function to invert a batch of time series of observation and target from the client's gradients:

$$\mathcal{L}_{\text{total}} = D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}) + \mathcal{R}_{TS}.$$

D. One-shot Target Reconstruction for $\mathcal{B} = 1$

In this section, we demonstrate how to reconstruct the target predictions of a fully connected layer f_{FC}, which in practice is the last layer of the forecasting model f. Given a fully connected layer f_{FC} with weights $\mathbf{W}$ and biases $\mathbf{b}$, and knowing the gradients $\nabla\mathbf{W}$ and $\nabla\mathbf{b}$ obtained with training inputs x and outputs y, we assume the objective function is the MSE $\mathcal{L}_{\text{MSE}} = \|\hat{y} - y\|_2$, where $\hat{y}$ are the layer's predictions. It is possible to reconstruct y one-shot for $\mathcal{B} = 1$ if $\nabla\mathbf{b}$ is invertible. The MSE loss is defined as: $\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$, where N is the number of elements. We can express the predicted output as $\hat{y} = \mathbf{W}x + \mathbf{b}$. The gradient with respect to the bias $\mathbf{b}$ and weights $\mathbf{W}$ are:

$$\nabla\mathbf{b} = \frac{2}{N}(\mathbf{W}x + \mathbf{b} - y) \tag{12}$$

$$\nabla\mathbf{W} = \frac{2}{N}(\mathbf{W}x + \mathbf{b} - y)x^T. \tag{13}$$

Equation (12) can be rearranged to solve for y:

$$y = \mathbf{W}x + \mathbf{b} - \nabla\mathbf{b}\cdot\frac{N}{2}. \tag{14}$$

Dataset	FREQ.	H	F	W	S_{attack}	S_{aux}
Electricity 370	15 MIN	96	96	192	96	4
London SmartMeter	30 MIN	48	48	96	48	2
Proprietary	15 MIN	96	96	192	96	4
KDDCup 2018	HOUR	120	48	168	24	1

TABLE IV: Dataset descriptions: sample frequency (FREQ.), observation length (O), future length (F), window size (W), and step size for attacked samples (S_{attack}) and for the auxilary samples (S_{aux}).

	Dataset	Electricity 370		London Smartmeter	
Model	Grad. Distance D	Input	Target	Input	Target
1	Cosine + L1	$0.083_{0.01}$	**9.3e-08**$_{0.00}$	$0.212_{0.03}$	**2.3e-06**$_{0.00}$
	Cosine + L2	$0.290_{0.17}$	**9.3e-08**$_{0.00}$	$0.481_{0.30}$	**2.3e-06**$_{0.00}$
	Cosine	$0.377_{0.19}$	**9.3e-08**$_{0.00}$	$0.860_{0.56}$	**2.3e-06**$_{0.00}$
	L2	$0.236_{0.05}$	**9.3e-08**$_{0.00}$	$0.783_{0.49}$	**2.3e-06**$_{0.00}$
	L1	**0.081**$_{0.03}$	**9.3e-08**$_{0.00}$	**0.208**$_{0.14}$	**2.3e-06**$_{0.00}$
2	Cosine + L1	**0.362**$_{0.14}$	**0.053**$_{0.04}$	$0.301_{0.20}$	$0.261_{0.29}$
	Cosine + L2	$0.628_{0.24}$	$0.119_{0.03}$	$0.816_{0.36}$	$0.602_{0.46}$
	Cosine	$0.619_{0.18}$	$0.630_{0.40}$	$1.049_{0.25}$	$0.920_{0.33}$
	L2	$0.629_{0.32}$	$0.613_{0.40}$	$0.869_{0.43}$	$0.860_{0.36}$
	L1	$0.471_{0.08}$	$0.095_{0.06}$	**0.278**$_{0.20}$	**0.244**$_{0.27}$

TABLE V: TS-Inverse without regularization for TCN model architecture results measured with sMAPE ↓. Grad. distance functions are: "Cosine + L1", "Cosine + L2", "Cosine", "L2", and "L1", for $\mathcal{B} \in [1, 2]$ and seeds (10, 43, & 28).

Here, x is unknown, but it can be reconstructed by substituting y from (14) into (13), we get:

$$\nabla \mathbf{W} = \frac{2}{N}\left(\mathbf{W}x + \mathbf{b} - \left(\mathbf{W}x + \mathbf{b} - \nabla \mathbf{b} \cdot \frac{N}{2}\right)\right) x^T$$
$$= \nabla \mathbf{b} \cdot x^T.$$

If $\nabla \mathbf{b}$ is invertible, such that $(\nabla \mathbf{b})^{-1}$ exists, then $\mathrm{x}^T = (\nabla \mathbf{b})^{-1} \nabla \mathbf{W}$. Finally, substituting x back into equation (14), we obtain:

$$y = \mathbf{W}(\nabla \mathbf{b})^{-1} \nabla \mathbf{W} + \mathbf{b} - \nabla \mathbf{b} \cdot \frac{N}{2}. \tag{15}$$

This demonstrates that the target y can be reconstructed in one shot for $\mathcal{B} = 1$, provided that $\nabla \mathbf{b}$ is invertible.

V. Evaluation

We first detail the experimental setup, followed by the attack accuracy of TS-Inverse, and an ablation study.

A. Experimental setup

1) Dataset: Four datasets are the Electricity (370)[1], KDDCup 2018 (Air Quality)[2], London Smartmeter[3], and a proprietary energy dataset. The energy datasets are sampled with 1 observation day and 1 future day to forecast. The KDDCup dataset has 5 observation days and 2 future days. The sample description of the datasets is summarized in Table IV. All datasets are normalized using the min-max method bounding it between 0 and 1.

[1]https://archive.ics.uci.edu/dataset/321/electricityloaddiagrams20112014
[2]https://zenodo.org/records/4656756
[3]https://zenodo.org/records/4656091

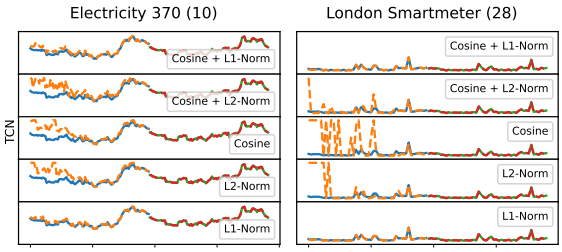


Fig. 6: TS-Inverse, reconstruction comparison on the TCN model architecture with different gradient distance functions and without regularization.

2) Data Pre-processing: We employ the rolling window method to preprocess the data, i.e., moving a window of fixed-width steps through the dataset, advancing by steps with each iteration. The data within this window is divided into observation and target segments $\mathbf{S}_{obs}$ and $\mathbf{S}_{tar}$ according to an observation length H and future length F. The result of the rolling window process is a dataset sample pairs of observations $\mathbf{S}_{obs}$ and targets $\mathbf{S}_{tar}$. The dataset-specific settings are described in Table IV. The datasets are divided into training, validation, and test sets with ratios of 64%, 16%, and 20%, respectively, by recursively applying a 20% split ratio. For the auxiliary dataset $\mathcal{D}_{aux}$, the validation set is used.

3) Baselines: The baselines that have been evaluated are the DLG-LBFGS [10], DLG-Adam [10], (InvG) [13], (DIA) [18], and (LTI) [19]. The implementation and configuration details are consistent with their open-sourced repositories. Where LTI is 250 epochs and others 5000 iterations.

4) Network Architectures: The FCN model has 3 FC layers with sigmoid acivation functions, the CNN model is based on LeNet with 1D Conv layers. The TCN architecture has a kernel size of 6, dilation factor of 2 and the number of levels is adjusted such that the receptive field is encompassing the observation sequence. The GRU-2-FCN and GRU-2-GRU both have the standard GRU implementation. All models apart from the GRU-2-GRU have a FC head that is outputting the final values. All hidden sizes of the architectures are 64.

The gradient inversion model f_{inv} has two modules for outputting observations and targets, each containing two residual blocks with hidden sizes of 768 and 512, respectively. It is trained for 75 epochs and the quantile levels are $\{0.1, 0.3, 0.7, 0.9\}$.

5) Evaluation Metrics: To measure the reconstruction quality the Symmetrical Mean Absolute Percentage Error (sMAPE ↓) is used, which is bounded between 0.0 and 2.0, and allows for better comparisons between datasets. When using other metrics, such as MSE and MAE, the ranks do not change.

B. Results overview for TS-Inverse

The comparison in Table III demonstrates that TS-Inverse outperforms other methods in time-series reconstruction across multiple datasets and architectures. Specifically, TS-Inverse achieves the lowest sMAPE values for both observation and target sequences. This is visually supported by Figure 5, which showcases individual reconstructions on various datasets, highlighting

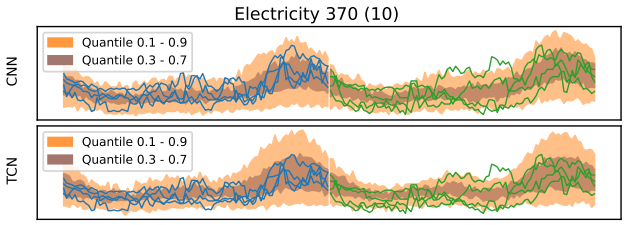


Fig. 7: TS-Inverse, Learned quantiles that function as prior in the learned regularization on Electricity 370 dataset for the CNN and TCN model architecture and $\mathcal{B} = 4$.

λ_Q^{obs}	0		0.1		0.5		1	
λ_Q^{tar}	Input	Target	Input	Target	Input	Target	Input	Target
0	$0.515_{0.11}$	$0.068_{0.05}$	$0.504_{0.12}$	$0.185_{0.11}$	$\mathbf{0.207}_{0.08}$	$0.061_{0.06}$	$0.203_{0.05}$	$0.060_{0.03}$
0.1	$0.567_{0.17}$	$\mathbf{0.048}_{0.01}$	$0.449_{0.14}$	$0.105_{0.05}$	$0.284_{0.02}$	$0.237_{0.11}$	$\mathbf{0.144}_{0.03}$	$\mathbf{0.024}_{0.02}$
0.5	$0.568_{0.18}$	$0.067_{0.02}$	$\mathbf{0.368}_{0.21}$	$\mathbf{0.026}_{0.02}$	$0.259_{0.08}$	$0.112_{0.08}$	$0.207_{0.04}$	$0.131_{0.06}$
1	$\mathbf{0.514}_{0.07}$	$0.053_{0.03}$	$0.522_{0.22}$	$0.342_{0.26}$	$0.226_{0.12}$	$\mathbf{0.040}_{0.03}$	$0.196_{0.04}$	$0.077_{0.03}$

TABLE VI: Results of different hyperparameters for quantile bounds regularization on observations (λ_Q^{obs}) and targets (λ_Q^{tar}) for the Electricity dataset with 370 seeds (10 and 43).

the effectiveness of TS-Inverse compared to baseline reconstructions in Figure 3. Additionally, the learned quantiles, depicted in Figure 7, highlight the method's capability to effectively capture and reconstruct quantile bounds for the FCN, CNN and TCN architectures with a batch size of 1 on the Electricity 370 dataset.

Effectiveness of the Proposed Gradient Distance Function

Table V presents the sMAPE results for different gradient distance functions without additional regularization, focusing on capturing the effects of the gradient distances alone. The gradient distance functions tested include Cosine + L1-Norm, Cosine + L2-Norm, Cosine, L2, and L1. The results indicate that distance functions incorporating the L1-Norm perform better with lower sMAPE results compared to other functions. This trend is consistent across various datasets and is particularly evident in the TCN architecture, as shown in Figure 6, which displays individual reconstructions for $\mathcal{B} = 1$. For the baseline architectures (FCN and CNN), the differences in sMAPE results among the distance functions are minimal.

Effectiveness of Regularization for Batch Reconstruction

The effects of the hyperparameters for the quantile bounds regularization are shown in Table VI. The hyperparameters for the quantile bounds regularization are optimized through a grid search over the values $\{0, 0.1, 0.5, 1\}$ for both the observation and target sequences as can be seen in Table VI. The results indicate that increasing the regularization term for observations leads to a decrease in the sMAPE metric, whereas the effect is less apparent for target sequences. From these results the optimal terms are found to be $\lambda_Q^{obs} = 1$ and $\lambda_Q^{tar} = 0.1$ for the Electricity 370 dataset. Subsequently, the effects of periodicity and trend regularization hyperparameters (λ_P and λ_T) are investigated, as detailed in Table VII. The values tested for both terms are $\{0.1, 0.5, 1\}$. The reconstructions incorporating all regularizations (periodicity, trend, and quantile bounds) for

	λ_P	0.5		1		2	
λ_T	Regularization	Input	Target	Input	Target	Input	Target
0.5	$\mathcal{R}_\mathcal{P} + \mathcal{R}_T$	$\mathbf{0.154}_{0.03}$	$\mathbf{0.118}_{0.10}$	$0.155_{0.05}$	$0.086_{0.06}$	$0.350_{0.14}$	$0.302_{0.18}$
	$\mathcal{R}_\mathcal{P} + \mathcal{R}_T + \mathcal{R}_Q$	$0.273_{0.12}$	$0.287_{0.19}$	$\mathbf{0.124}_{0.00}$	$\mathbf{0.039}_{0.03}$	$\mathbf{0.174}_{0.02}$	$\mathbf{0.199}_{0.01}$
1	$\mathcal{R}_\mathcal{P} + \mathcal{R}_T$	$0.197_{0.04}$	$0.075_{0.02}$	$0.350_{0.20}$	$0.330_{0.29}$	$0.162_{0.01}$	$\mathbf{0.081}_{0.01}$
	$\mathcal{R}_\mathcal{P} + \mathcal{R}_T + \mathcal{R}_Q$	$\mathbf{0.161}_{0.01}$	$\mathbf{0.059}_{0.02}$	$\mathbf{0.141}_{0.01}$	$\mathbf{0.080}_{0.05}$	$\mathbf{0.158}_{0.01}$	$0.098_{0.03}$
2	$\mathcal{R}_\mathcal{P} + \mathcal{R}_T$	$0.217_{0.02}$	$\mathbf{0.058}_{0.01}$	$0.184_{0.02}$	$0.102_{0.01}$	$0.314_{0.15}$	$0.254_{0.21}$
	$\mathcal{R}_\mathcal{P} + \mathcal{R}_T + \mathcal{R}_Q$	$\mathbf{0.184}_{0.02}$	$0.082_{0.02}$	$\mathbf{0.132}_{0.02}$	$\mathbf{0.051}_{0.04}$	$\mathbf{0.124}_{0.03}$	$\mathbf{0.114}_{0.09}$

TABLE VII: The results of using the periodicity and trend regularization together, with or without including the the quantile bounds regularization technique. Dataset Electricity 370 with seeds (10 and 43)

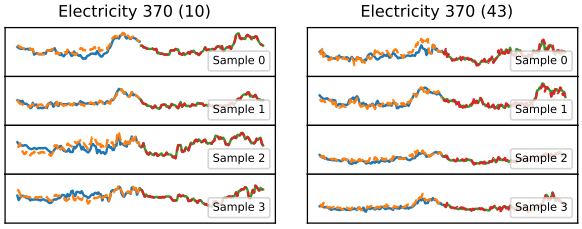


Fig. 8: TS-Inverse, reconstructions with all regularization applied for the TCN model architecture and $\mathcal{B} = 4$.

the TCN model with a batch size of 4 are illustrated in Figure 8. We note that bigger the batch sizes are, more challenging the inversion is.

C. Defenses

We investigate a defense strategy using GRU in the forecasting model to counter TS-Inverse. GRU-based forecasting models are more robust against attacks due to the many-to-one mapping caused by the recurrent design. We attack the GRU-2-FCN and GRU-2-GRU models, with reconstruction results shown in Figure 9. The results show that the targets associated with the FCN part remain vulnerable. We hypothesize that regularization causes the values to transfer to the observations, as the observations of the GRU-2-FCN architecture closely resemble the targets. This is backed by the results of employing a GRU-2-GRU architecture, which prevents detailed reconstruction of the observations and targets. However, the attack does seem to be able to extract the trend from the gradients. Overall, we the GRU-based architectures enhance the robustness against privacy attacks because of the many-to-one mapping design.

VI. Conclusion

This study presents the first empirical analysis and effective algorithm, TS-Inverse, of reconstructing TS data for

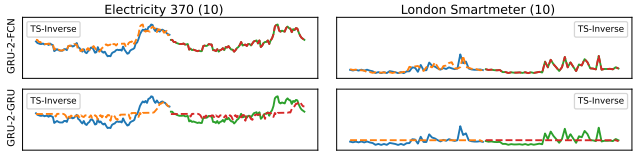


Fig. 9: TS-Inverse reconstructions on Electricity 370 and London Smartmeter datasets for the GRU-2-FCN and GRU-2-GRU architectures $\mathcal{B} = 1$.

federated TSF models. Our empirical analysis demonstrates that existing inversion methods, developed for image and text classification, are less effective for TSF due to the mismatch between gradient distance and the gradient magnitudes and TS-specific model architecture, e.g., TCN & GRU. To address these challenges, we propose `TS-Inverse`, which leverages a gradient inversion model and TS-specific characteristics as regularization during reconstruction. Specifically, the proposed TS-regularization include periodicity and trend. Furthermore, the gradient inversion model is trained with an auxiliary dataset to map gradients to quantile bounds of observations and targets. These bounds are also used to regularize the reconstruction. We extensively evaluate `TS-Inverse` on four data sets against five baselines, showing a 2x-10x improvement in reconstruction sMAPE and indicating the potential of GRU-based architecture in defending against inversion attacks.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. AISTATS, PMLR, 2017, pp. 1273–1282.

[2] J. Zhu, H. Dong, W. Zheng, S. Li, Y. Huang, and L. Xi, "Review and prospect of data-driven techniques for load forecasting in integrated energy systems," *Applied Energy*, vol. 321, p. 119 269, 2022, ISSN: 0306-2619.

[3] N. Ahmad, Y. Ghadi, M. Adnan, and M. Ali, "Load forecasting techniques for power system: Research challenges and survey," *IEEE Access*, vol. 10, pp. 71 054–71 090, 2022, ISSN: 2169-3536.

[4] S. Čaušević, S. Sharma, S. B. Aziza, A. van der Veen, and E. Lazovik, "Lv grid state estimation using local flexible assets: A federated learning approach," in *27th International Conference on Electricity Distribution (CIRED 2023)*, 2023, pp. 1045–1049.

[5] S. Čaušević, R. Snijders, G. Pingen, and et al., "Flexibility prediction in smart grids: Making a case for federated learning," in *CIRED 2021 - The 26th International Conference and Exhibition on Electricity Distribution*, 2021, pp. 1983–1987.

[6] N. Truong, K. Sun, S. Wang, F. Guitton, and Y. Guo, "Privacy preservation in federated learning: An insightful survey from the gdpr perspective," *Computers Security*, vol. 110, p. 102 402, 2021, ISSN: 0167-4048.

[7] L. Lyu, H. Yu, X. Ma, and et al., "Privacy and robustness in federated learning: Attacks and defenses," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2022, ISSN: 2162-2388.

[8] N. Rodríguez-Barroso, D. Jiménez-López, M. V. Luzón, F. Herrera, and E. Martínez-Cámara, "Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges," *Information Fusion*, vol. 90, pp. 148–173, 2023, ISSN: 1566-2535.

[9] Y. Huang, S. Gupta, Z. Song, K. Li, and S. Arora, "Evaluating gradient inversion attacks and defenses in federated learning," in *Advances in Neural Information Processing Systems*, vol. 34, Curran Associates, Inc., 2021, pp. 7232–7241.

[10] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems*, vol. 32, Curran Associates, Inc., 2019.

[11] Y. Wang, J. Deng, D. Guo, and et al., "Sapag: A self-adaptive privacy attack from gradients," *arXiv:2009.06228 [cs, stat]*, Sep. 2020.

[12] B. Zhao, K. R. Mopuri, and H. Bilen, "Idlg: Improved deep leakage from gradients," *arXiv:2001.02610 [cs, stat]*, Jan. 2020.

[13] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients – how easy is it to break privacy in federated learning?" *arXiv:2003.14053 [cs]*, Sep. 2020.

[14] J. Deng, Y. Wang, J. Li, and et al., "Tag: Gradient attack on transformer-based language models," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds., Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 3600–3610.

[15] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," *arXiv:2104.07586 [cs]*, Apr. 2021.

[16] J. Jeon, J. Kim, K. Lee, S. Oh, and J. Ok, "Gradient inversion with generative image prior," *arXiv:2110.14962 [cs]*, Oct. 2021.

[17] J. Geng, Y. Mou, F. Li, and et al., "Towards general deep leakage in federated learning," *arXiv:2110.09074 [cs]*, Jan. 2022.

[18] D. Scheliga, P. Mäder, and M. Seeland, "Dropout is not all you need to prevent gradient leakage," *arXiv:2208.06163 [cs]*, Nov. 2022.

[19] R. Wu, X. Chen, C. Guo, and K. Q. Weinberger, "Learning to invert: Simple adaptive attacks for gradient inversion in federated learning," in *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, PMLR, Jul. 2023, pp. 2293–2303.

[20] K. Cho, B. van Merrienboer, C. Gulcehre, and et al., "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv:1406.1078 [cs, stat]*, Sep. 2014.

[21] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv:1803.01271 [cs]*, Apr. 2018.

[22] M. N. Fekri, H. Patel, K. Grolinger, and V. Sharma, "Deep learning for load forecasting with smart meter data: Online adaptive recurrent neural network," *Applied Energy*, vol. 282, p. 116 177, Jan. 2021, ISSN: 0306-2619.

[23] M. Savi and F. Olivadese, "Short-term energy consumption forecasting at the edge: A federated learning approach," *IEEE Access*, vol. 9, pp. 95 949–95 969, 2021, ISSN: 2169-3536.

[24] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. John Wiley Sons, May 2015, ISBN: 978-1-118-67492-5.

[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986, ISSN: 1476-4687.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, ISSN: 0899-7667.

[27] M. Alhussein, K. Aurangzeb, and S. I. Haider, "Hybrid cnn-lstm model for short-term individual household load forecasting," *IEEE Access*, vol. 8, pp. 180 544–180 557, 2020, ISSN: 2169-3536.

[28] P. Lara-Benítez, M. Carranza-García, J. M. Luna-Romera, and J. C. Riquelme, "Temporal convolutional networks applied to energy-related time series forecasting," *Applied Sciences*, vol. 10, no. 7, p. 2322, Jan. 2020, ISSN: 2076-3417.

[29] H. Bousbiat, R. Bousselidj, Y. Himeur, and et al., "Crossing roads of federated learning and smart grids: Overview, challenges, and perspectives," *arXiv:2304.08602 [cs]*, Apr. 2023.

[30] J. D. Fernández, S. P. Menci, C. M. Lee, A. Rieger, and G. Fridgen, "Privacy-preserving federated learning for residential short-term load forecasting," *Applied Energy*, vol. 326, p. 119 915, Nov. 2022, ISSN: 0306-2619.

[31] E. U. Soykan, Z. Bilgin, M. A. Ersoy, and E. Tomur, "Differentially private deep learning for load forecasting on smart grid," in *2019 IEEE Globecom Workshops (GC Wkshps)*, Dec. 2019, pp. 1–6.

[32] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *2019 IEEE Symposium on Security and Privacy (SP)*, May 2019, pp. 739–753.

[33] F. Mo, A. Borovykh, M. Malekzadeh, H. Haddadi, and S. Demetriou, "Layer-wise characterization of latent information leakage in federated learning," *arXiv:2010.08762 [cs]*, May 2021.

[34] Y. Zheng, "Dropout against deep leakage from gradients," *arXiv:2108.11106 [cs]*, Nov. 2022.

[35] Z. Jiang, J. Gu, M. Liu, and D. Z. Pan, "Delving into effective gradient matching for dataset condensation," *arXiv:2208.00311 [cs]*, Jul. 2022.

[36] T. Zhang, Y. Zhang, K. Wang, and et al., "Two trades is not baffled: Condensing graph via crafting rational gradient matching," *arXiv:2402.04924 [cs]*, Feb. 2024.

Chapter 3

Background

This chapter provides additional background information for concepts used in the paper, such as federated learning, time series, time series forecasting, privacy risks in federated learning, and gradient inversion attacks.

If at any point in the paper you find yourself wondering about any of these topics, this chapter will provide additional information and insights. This chapter also offers more detail on closely related topics. Furthermore, it provides formalizations that can help you understand the concepts in a more structured way.

In Section 3-1, we describe the two concepts of horizontal and vertical federated learning and which is used in the paper and use case. Furthermore, the paper provides a minimal formalization of federated learning for its methodology. Here we provide a more detailed formalization and more detail on the FedSGD strategy and an alternative (FedAvg) strategy, highlighting their differences and similarities.

If you are not familiar with time series and time series forecasting, Section 3-2 will go into more detail and discuss characteristics that can be present in time series, two of which are used in the paper. Furthermore, a more general formalization of time series forecasting is given: We describe the differences between single-step and multi-step forecasting and between univariate and multivariate time series. We also provide insights into additional data used in practical time series forecasting scenarios. This section ends by explaining how these time series concepts relate to load forecasting and their related work.

In Section 3-3, we examine the intersection of federated learning and time series forecasting, mainly its challenge with regard to non-independent and identically distributed data, which is also a challenge in load forecasting. This concept is explained as it is a widely researched topic in load forecasting.

In Section 3-4, we provide background information about the various types of privacy risks associated with federated learning. We introduce the European General Data Protection Regulation (GDPR) law as a motivator for researching privacy risks in this context. Additionally,

we assume an "honest-but-curious" server-side attack in our paper and discuss alternative scenarios. We explain why we chose the "honest-but-curious" server-side attack assumption for the load forecasting use case.

Lastly, in Section 3-5, we provide more details about the related gradient inversion attacks mentioned in the paper.

3-1 Federated Learning

Federated Learning is a distributed machine learning approach that enables multiple clients (e.g., mobile devices, organizations, households) to collaboratively train a model while keeping their data localized [12]. The model is trained under the coordination of a central server. Within Federated Learning a distinction is made between Horizontal Federated Learning and Vertical Federated Learning, which will be differentiated in the first section. Hereafter in the thesis we will focus only on Horizontal Federated Learning, and will be referred to as Federated learning. In the subsequent section the formalization of Federated learning is given, and the two strategies that were first proposed by [12] are elaborated.

3-1-1 Horizontal vs Vertical Federated Learning

Horizontal federated learning and vertical federated learning differ primarily in the distribution of data across participating clients [17]. In horizontal federated learning, also known as sample-based federated learning, each entity has data that includes the same features but different samples. This is typical when organizations operating in the same domain, like hospitals, collaborate without sharing patient data. On the other hand, vertical federated learning, or feature-based federated learning, involves clients that share data on the same samples but with different features. This scenario is common when clients such as a bank and an insurance company, which serve the same customers, collaborate without exposing sensitive data. These distinctions allow federated learning to be applied flexibly depending on data distribution and privacy requirements. Hereafter, federated learning is revering to the case of horizontal federated learning.

In the context of of low voltage load forecasting, where each client, such as households or businesses, possesses only their own private energy consumption data. In this scenario, while the domain remains consistent—energy consumption—the data distributions can vary significantly from client to client due to differing usage patterns, making it a unique horizontal federated learning challenge.

3-1-2 Formalization of Federated Learning

A federated learning framework involves K clients, denoted as $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$. In this framework, data is divided among these clients, leading to the creation of data silos $\{D_1, D_2, \ldots, D_K\}$. Each data silo D_k contains pairs of input and output data $\left\{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\right\}_{i=1}^{n_k}$, where k represents the specific client and for simplicity, k ranges from 1 to K. The variable n_k denotes the number of data points in D_k.

The clients jointly train a global model via a strategy determined by the server. The two typical strategies introduced are Federated Averaging (FedAvg) and Federated Stochastic Gradient Descent (FedSGD) [12].

Both strategies can be expressed in a generalized way where each client receives the global model $\mathbf{W}_g^r$ from the server where g indicates its the global model, and r the global training round. Each client $\mathcal{C}_k$ train the model locally using D_k resulting in a new model $\mathbf{W}_k^{r+1}$. The locally trained models $\mathbf{W}_k^{r+1}$ can be viewed as a function of the global model $\mathbf{W}_g^r$ and the local data.

Both FedAvg and FedSGD strategies can be generalized to express the update as:

$$\mathbf{W}_k^{r+1} = \mathbf{W}_g^r - \eta_k \nabla \mathcal{L}(\mathbf{W}_g^r, \mathcal{B}_k), \tag{3-1}$$

where η_k is a learning rate that encapsulates the difference in the amount of local training between FedAvg and FedSGD, and $\nabla \mathcal{L}(\mathbf{W}_g^r, \mathcal{B}_k)$ is the gradient of the loss function $\mathcal{L}$ with respect to the model parameters using the local data $\mathcal{B}_k$. The learning rate η_k and local data $\mathcal{B}_k$ varies depending on the strategy used:

Federated Averaging (FedAvg): Each client performs multiple local updates over E epochs. The local data $\mathcal{B}_k$ is in this case the dataset $\mathcal{D}_k$. Within each epoch, there are multiple gradient steps. Let G denote the number of gradient steps per epoch, which is a function of the batch size and n_k:

$$G = \left\lceil \frac{n_k}{\text{batch size}} \right\rceil \tag{3-2}$$

The effective learning rate η_k in this case is:

$$\eta_k \approx EG\alpha, \tag{3-3}$$

where E is the number of local epochs, G is the number of gradient steps per epoch, and α is the local learning rate. This approximation assumes that the local model parameters do not change to much during the each local training step.

Federated Stochastic Gradient Descent (FedSGD): Each client performs a single local update on a mini-batch $\mathcal{B}_k \in \mathcal{D}_k$ and immediately sends the updated parameters back to the server. The effective learning rate η_k in this case is simply the local learning rate α:

$$\eta_k = \alpha. \tag{3-4}$$

The updated model $\mathbf{W}_k^{r+1}$ is sent back to the server which aggregates it together with the other clients' updates according to each clients' amount of data resulting in $\mathbf{W}_g^{r+1}$. This updated global model $\mathbf{W}_g^{r+1}$ is computed as follows:

$$\mathbf{W}_g^{r+1} = \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{W}_k^{r+1}, \tag{3-5}$$

where n_k is the number of data points in client k's data $\mathcal{D}_k$, and $n = \sum_{k=1}^{K} n_k$ is the total number of data samples across all clients.

3-2 Time Series and Time Series Forecasting

Time series forecasting involves predicting future values based on previously observed values. A time series is a sequence of data points collected or recorded at regular time intervals. Time series data captures the dynamics and temporal dependencies inherent in the data [18]. Time series forecasting is critical in numerous applications, such as finance, weather prediction, and load forecasting [19]. First we will describe time series and its characteristics and then we will describe the process of forecasting time series and which methods are used for this.

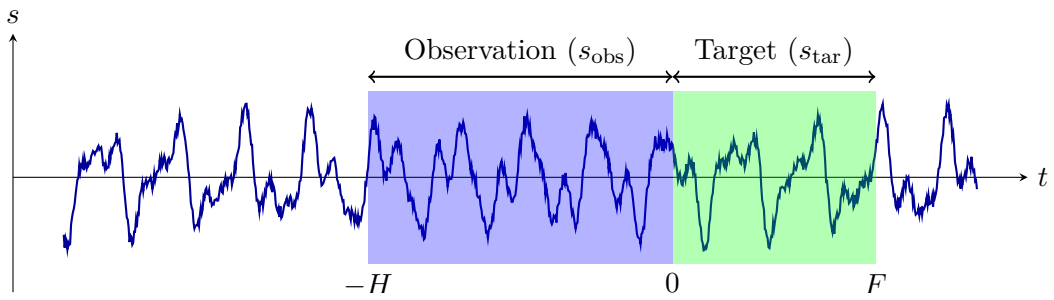


Figure 3-1: Example of univariate time-series forecasting situation.

Time Series Data

Time series data is characterized by its sequential nature, with observations indexed by time. Each data point in a time series can be represented as $\mathbf{s}_t$, where t denotes the time index. Key characteristics of time series include trend, periodicity, cyclic patterns, and irregular fluctuations [18]. Trends indicate long-term progression, such as the upward trend in global temperatures over decades. Periodicity refers to regular repeating patterns, such as daily cycles (e.g., day and night) and weekly cycles (e.g., increased restaurant visits on weekends). Seasonality, a specific type of periodicity, refers to patterns tied to the calendar, such as annual cycles (e.g., increased retail sales in December) and monthly cycles (e.g., higher electricity consumption in summer and winter). Cyclic variations represent patterns that occur at irregular, long-term intervals, such as economic cycles of recession and expansion. Irregular components reflect random noise. Understanding these components is essential for accurate modeling and forecasting, as they provide insight into the underlying processes driving the observed data.

Formalization of Time Series Forecasting

We begin with a simple scenario of time series forecasting and gradually introduce more complex concepts. We start with single step forecasting, then introduce multi-step forecasting, followed by introducing univariate vs. multivariate time series, and at last the incorporation of second hand predictions into the conditions to forecast.

Observations and Targets: The observations are synonyms for the model inputs ($\mathbf{x}$) and targets for ground-truth outputs ($\mathbf{y}$). Consider a historical time range, consisting of H steps, as the set of observations obs $= \{t \in \mathbb{Z} \mid -H < t \leq 0\}$. Now consider a historical time series

$s_{\text{obs}} = \{\mathbf{s}_{-H+1}, \mathbf{s}_{-H+2}, \ldots, \mathbf{s}_0\}$ which is used to predict the next value $\mathbf{s}_{t+1}$. This is known as single-step forecasting, where only the immediate next value is predicted. In many practical applications, we need to predict multiple future values at once, this leads to multi-step forecasting. Now, given the observation time series $\mathbf{s}_{\text{obs}}$, the objective is to forecast a future range of F steps into the future, a set of targets $s_{\text{tar}} = \{t \in \mathbb{Z} \mid 0 < t \leq F\}$ The goal is to forecast the future values $\mathbf{s}_{\text{tar}} = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_F\}$ as illustrated in Figure 3-1.

Univariate vs. Multivariate Time Series: Now consider an additional dimension in the time series observations which can be represented as $\mathbf{S}_{\text{obs}} \in \mathbb{R}^{H \times d}$, where d denotes the number of distinct features or dimensions in the dataset. Similarly, the target forecast data can be represented as $\mathbf{S}_{\text{tar}} \in \mathbb{R}^{F \times d}$. A time series is categorized as univariate when $d = 1$, as depicted in Figure 3-1, and as multivariate when $d > 1$. In a univariate time series, each time step t is associated with a single value $\mathbf{s}_t$. In contrast, in a multivariate time series, each time step t is associated with a vector of values $\mathbf{s}_t = \{\mathbf{s}_{t,1}, \ldots, \mathbf{s}_{t,d}\}$, where d is the number of features.

External Forecasts and Static Covariates: In more advanced forecasting scenarios, external forecasts can serve as additional inputs, improving the accuracy of the predictions [20]. For example, using detailed weather forecasts for the next day can significantly enhance the accuracy of energy load predictions for that day. Additionally, static covariates, such as location or demographic information, can also be used to further refine the forecasting models. Although this approach is beyond the scope of this research, it is important to acknowledge its potential benefits for forecasting.

3-2-1 Forecasting Models

Traditional statistical methods, such as Autoregressive Integrated Moving Average (ARIMA) [18], leverage linear relationships in the data to make predictions. However, these methods often fall short in capturing complex nonlinear patterns.

Deep learning approaches, particularly Recurrent Neural Networks (RNNs) [21] and their variants like Long Short-Term Memory (LSTM) [22] networks and Gated Recurrent Units(GRUs) [23], have demonstrated superior performance in time series forecasting. These models can capture intricate temporal dependencies and long-term relationships within the data. LSTMs address the vanishing gradient problem in standard RNNs by incorporating memory cells that retain information over long periods, while GRUs simplify the LSTM architecture by combining the cell state and hidden state.

Another powerful deep learning architecture is the Temporal Convolutional Network (TCN) [24], which leverages causal convolutions and dilation to model temporal dependencies. TCNs have shown increased performance over RNN-based models due to their ability to parallelize training and handle long sequences effectively [24].

3-2-2 Low Voltage Load Data and Forecasting

Low voltage load forecasting is the process of predicting future electricity demand at the lower end of the voltage spectrum, typically associated with residential or small commercial consumers [3, 7]. This type of forecasting plays a critical role in smart grids by ensuring a

balance between energy demand and supply [2]. Accurately predicting low voltage load is essential for optimizing energy usage, integrating renewable energy sources, and managing grid stability, particularly with the increasing adoption of decentralized energy systems and smart grids [25].

Time Series Characteristics of Low Voltage Load Data

Low voltage load data, which captures the electricity consumption of residential or small commercial consumers, exhibits distinct time series characteristics that are critical for effective grid management [26].

Trend: Household load data may display a trend reflecting long-term changes in electricity usage. This could be influenced by factors such the number of household members, changes in household appliances, or shifts towards energy-efficient technologies. Identifying the trend component $\mathbf{S}_{\text{trend}}$ in load data is crucial for understanding the general direction of electricity consumption over time.

Periodicity: Periodicity in low voltage load data is evident through periodic fluctuations corresponding to regular intervals such as daily or weekly cycles. For instance:

- **Daily Periodicity:** Consumption patterns typically show higher usage during evening hours when residents are home and lower usage during the night and early morning hours.
- **Weekly Periodicity:** Weekends often exhibit different consumption patterns compared to weekdays, often with higher usage on weekends due to more people being at home.
- **Annual Periodicity:** Electricity usage can also vary across seasons, with higher consumption in the winter (due to heating), and some households or regions in the summer (due to cooling).

The periodic component captures these regular patterns and is crucial for accurate load forecasting. These periodic patterns are formalized with a period p, which indicates the number of time steps (such as hours, days, or weeks) after which the cyclic behavior repeats itself. Assuming an hourly sample rate, for example, if $p = 24$, the pattern repeats every 24 hours, indicating daily periodicity. Similarly, if $p = 168$, the pattern repeats every 168 hours (7 days), indicating weekly periodicity.

Cyclic Patterns: Unlike seasonality, cyclic patterns in household load data occur at irregular intervals and can be associated with broader economic or social cycles. For example, economic downturns or periods of increased remote work (e.g., during a pandemic) can lead to shifts in typical electricity consumption patterns. The cyclic component helps to identify these non-regular fluctuations. To formalize the cyclic pattern occurring at irregular intervals, one approach is to introduce an additional indicator in the form of a second time series that correlates with the pattern, increasing d, such as an economic wellbeing indicator.

Irregular Fluctuations: Household load data also contains irregular or random components, representing short-term anomalies or noise that cannot be attributed to trend, seasonality, or cyclic patterns. These could be due to unexpected events such as power outages, extreme weather conditions, or atypical usage patterns on specific days.

Deep Learning Load Forecasting

The proliferation of smart meters has enabled the collection of high-resolution consumption data, paving the way for data-driven models. Among these, deep learning models, such as LSTM, GRU, CNN and TCN based architectures, have shown significant promise, [6, 4, 27, 28, 5, 29]. These models can learn complex patterns from historical data without needing explicit physical information.

Training Objectives in Load Forecasting

In load forecasting, standard loss functions such as Mean Squared Error (MSE) and Mean Absolute Error (MAE) are widely used. MSE is known to converge to the mean load, making it effective for capturing average load patterns. On the other hand, MAE converges to the median load, which can be more robust to outliers. However, these loss functions do not specifically target the accurate prediction of load peaks, which are critical for efficient grid management and optimization.

To address this, alternative loss functions like Dynamic Time Warping (DTW) [30, 31] and its differentiable variant, soft-DTW [32], have been explored [33]. DTW focuses on the similarity of time series shapes, offering potential improvements in aligning predicted and actual load curves. Soft-DTW extends this by providing a smooth and differentiable approximation, making it suitable for training with gradient descent. These methods emphasize the overall shape similarity, which can lead to better peak load forecasting performance, enhancing grid reliability and efficiency [33]. While peak load prediction is of significant interest for load forecasting and grid optimization [7], it is not investigated further in this thesis because it is outside the scope.

3-3 Federated Time Series Forecasting

In the context of federated learning, time series forecasting can be formalized such that each client's $\mathcal{C}_k$ data silo $\mathcal{D}_k$ contains at least one time series $\mathbf{S}_k$. These time series across clients represent the same feature. Without federated learning, each client can only train a forecasting model on its own data, limiting the model's generalizability due to the restricted amount of data [34, 35].

Federated learning alleviates this by assuming that the data distributions among clients are diverse [36, 37]. This diversity ensures the global model captures a wide range of patterns and anomalies present in different time series, leading to better performance on new, unseen data [36, 37]. This makes federated learning a powerful approach for time series forecasting across various domains.

3-3-1 Non-Independent and Identically Distributed Data

However, too much diversity can have negative effects on the convergence of a federated forecasting model [38]. If the diversity of the data is too high among clients, it is formally termed non-Independent and Identically Distributed (non-IID) data. This refers to data

where individual samples are correlated with each other and are not drawn from the same probability distribution. Correlation is detrimental because the model may overfit to the patterns and noise specific to those correlations, failing to capture the true underlying data distribution. Data not originating from the same distribution is problematic because it can become too complex for the model to properly learn a unified representation. The diverse patterns can be too conflicting, leading to insufficient generalization.

3-3-2 Non-IIDness in Federated Load forecasting

In the context of federated low voltage load forecasting, non-IIDness arises due to correlations and differences in data distributions across various clients [38, 39, 40, 41, 42].

The different data distributions stem from the fact that each client's data represents unique load consumption patterns influenced by lifestyle choices, number of household members, or for an an industrial or business related building, the opening hours or types of services all uniquely influence the load patterns. The data correlations can for instance be caused by regional factors such as all being of similar socioeconomic status or dealing with the same temperature and weather patterns. These factors create correlations within the data of each client, as the consumption patterns are not random but are driven by region-specific behaviors [41, 42]. For instance, a model trained predominantly on industrial load patterns may not perform well on residential data, and vice versa.

To address these issues, strategies such as personalized federated learning can be employed, allowing each client to maintain a model that is fine-tuned to its local data while still benefiting from the global model [34, 40, 43, 44]. Clustered federated learning can also be useful, grouping clients with similar distributions to group the diversity into seperate models [45, 46, 39, 41]. These approaches help mitigate the impact of non-IIDness, leading to more accurate and robust load forecasting models. Even though the convergence issues that arise from the non-IID data among the federated clients is outside the scope of rest of the thesis, it is important to note that each individual load profile is private and contains unique information.

3-4 Privacy Risks for Federated Learning

In the context of federated learning, a "privacy risk" refers to the potential threats and vulnerabilities that can lead to the exposure of sensitive information during the collaborative training process [14]. Privacy risks in federated learning arise when an adversary, either within or outside the system, is able to infer sensitive data from the clients' updates or model parameters exchanged during the training process [14]. These risks are primarily due to the shared nature of federated learning, where the shared gradients or model updates can leak private information about the training data of individual clients. Understanding these privacy risks is crucial, especially when considering the stringent data protection requirements imposed by regulations such as the General Data Protection Regulation (GDPR) [47].

3-4-1 GDPR and Federated Learning

The General Data Protection Regulation (GDPR) [47] is a comprehensive data protection law that came into effect in the European Union in 2018, designed to safeguard personal

data and ensure privacy rights for individuals. It mandates strict data privacy and security measures, significantly impacting technologies like federated learning, which is particularly suitable for keeping training data local and private [8]. By decentralizing data processing, federated learning aligns well with GDPR's principles of data privacy and security, as it avoids central data storage risks [8].

However, despite federated learning's alignment with GDPR principles by keeping data local and reducing central data storage risks, significant privacy challenges remain [14]. Adversaries may exploit the shared updates to infer private data with inference and reconstruction attacks [8]. Investigating these attacks from an attacker's perspective will enhance our understanding of their implications and help develop more effective countermeasures.

3-4-2 Client or Server-side Attacks

In federated learning, privacy risks can manifest through attacks either on the client-side or the server-side, each presenting unique challenges and vulnerabilities. The main difference lies in the amount of information that is available to the adversary. Client-side attacks have knowledge of the aggregated global model sent by the server, the locally trained model, and its own private dataset [15]. The server on the other hand has access to the aggregated model, the trained models or their gradients sent back by the clients, the identifiers of the clients, the size of their dataset [15]. There is also the outsider-side, which only has access to the outputs of the trained global model [15], however, together with the client-side adversaries, these are outside the scope of this thesis.

3-4-3 Honest-but-curious vs. Malicious Attacks

In federated learning, attacks can also be categorized into honest-but-curious and malicious, or in other words active and passive [15]. Active attackers interfere with the training process, by sharing modified model updates or gradients. For instance, they actively modify the model in such a way that they can train it to output a specific value given a specific input, also called a backdoor attack. A passive, attacker adheres to the federated protocols and shares the honest model updates. Their goal is mostly to obtain private information about other clients. In this thesis, we focus on the honest-but-curious type.

3-4-4 Types of Privacy Attacks

In this section we will briefly introduce the different types of privacy attacks from a server-side perspective. The common types of attacks are membership inference, property inference, and data reconstruction.

Membership Inference: Membership inference attacks in federated learning determine whether a specific data point was used in the training set [48, 49]. This type of attack is particularly applicable in classification tasks where the adversary aims to identify the presence of sensitive data, such as medical records or personal profiles, within the training dataset. The primary assumption is that the attacker has some auxiliary knowledge about the data distribution or access to similar data points used during training [14, 48]. Through

the observation of model updates or gradients, the adversary can infer the membership status of data points.

Property Inference: Property inference attacks in federated learning target the extraction of specific attributes from clients' training data that are not directly related to the primary learning task [50, 51, 52]. These attacks are applicable to deep learning tasks such as image classification. The key assumption is that the adversary has access to auxiliary training data correctly labeled with the target property and can observe the gradients. These attacks can leak sensitive information such as the presence of certain features or attributes in the training data, potentially revealing private aspects of the data, like the presence of a specific object in images or particular words in text data [14].

Data Reconstruction: These are the core types of attacks experimented with on time series data in this thesis. All of the existing literature either evaluates these attacks on images [53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63], text [64], or tabular [65] data and none on time series. We will briefly introduce the concept here, but also have a separate section going deeper into the related works. Data reconstruction attacks in federated learning involve an adversary extracting original training data and their associated labels from shared gradients. The attacks leverage the fact that gradients are derived from specific data points, and by analysing these gradients, an attacker can reconstruct the original inputs and labels. The key assumption is that the attacker has access to the model parameters and its gradients shared during the training process [53].

3-4-5 Privacy Risks for Federated Load Forecasting

In this section we will discuss the implications of the different attacks and assumptions in the context of load forecasting. The paper and research was focused on server-side honest-but-curious data recreation attacks.

Server-side Attack: We will discuss third-party adversaries, which are not directly involved in the federated system, adversarial clients that attempt to steal private information from other clients, or adversarial servers that steal information about its connected clients.

The fair assumption can be made that third-party adversaries do not have direct access to the federated learning system unless they hack into the smart meters or the server. If the adversary is able to compromise the smart meter, they might as well steal the private data directly, bypassing the federated learning process entirely. In this case it is more practical to enhance the security of the smart meters rather than modifying the fundamentals of the federated learning process itself. On the other hand, if they hack the server, attackers would still need to exploit server-side attacks in order to retrieve private information.

Adversarial clients attempting to infer private information about other clients in the federated system present a different challenge. These types of privacy attacks are not yet fully developed, and client-side attacks are often malicious in an attempt to modify the global model in a particular way [15]. Overall, from the perspective of understanding privacy risks within the federated learning setup, server-side attacks are of most interest and pose the biggest risk as they have direct access to the learning process and have more knowledge of the federated process as had been described in Subsection 3-4-2.

Honest-but-curious Server: We focus on the honest-but-curious case because, as we focus on server-side attacks, and if these servers would be malicious and tamper with the training protocols, they would risk repercussions or fines through the GDPR law. So it is reasonable to assume the servers would not act maliciously. We are interested in amount of training data that is actually present in the shared model updates.

Type of privacy attack: We will discuss the types of attacks that were mentioned in **??**. With membership inference attacks an adversarial server can attempt to determine whether a specific time series sequence was used in the training set. However, for load forecasting, this information is less critical compared to other domains for a couple of reasons. In image or text domain, the adversary can check for known samples, such as faces, or specific personal information. In load forecasting, each client only contains its own data, not that of multiple individuals. In this situation, checking for membership makes less sense as it would translate more to a data recreation attack.

Furthermore, property inference attacks, attempt to infer properties or patterns that is not directly related to the task. For example, identifying whether a household is unoccupied. However, such inferences may be gathered through other means more effectively.

Form the GDPR mandated guidelines, it is most important that the training data cannot be revealed during the training process [8]. From this perspective, the most severe privacy threat is data reconstruction. Gradient inversion attacks can allow the server to reconstruct the original time series data, thereby breaching the privacy that federated learning aims to protect [15].

3-5 Related Studies: Gradient Inversion Attacks

In this section we will go into depth of the related studies of gradient inversion attacks, as described in Section 3-5. In the following section we assume that the gradients stem from the Federated Stochastic Gradient Decent strategy as described in Section 3-1. Some related work also perform gradient inversion attacks on models trained with the Federated Averaging strategy which will be highlighted.

Each of the attacks attempt to reconstruct the exact training input and label for a given gradient using gradient decent. The modalities that are being attacked are images, text and tabular data during classification tasks.

It is important to know that gradients are aggregated over all of the samples. So, while reconstructing gradients of a single sample, or batch size of 1, is straight forward, the reconstruction of multiple samples or larger batches is more challenging.

A large portion of the attacks work by optimizing randomly initialized dummy inputs and labels. They are optimized by minimizing the distance between these dummy gradients, which are calculated using the dummy inputs, labels and model parameters, and the gradients that are "leaked" during the federated learning process.

More formally, the local training data $\mathcal{B}_k$ from Section 3-1 consists of training inputs ($\mathbf{x}$) and labels ($\mathbf{y}$), and the gradients $\nabla\mathcal{L}(\mathbf{W}_g^r, \mathcal{B}_k)$ are derived as:

$$\nabla\mathcal{L}(\mathbf{W}_g^r, \mathcal{B}_k) = \frac{\partial\mathcal{L}(F(\mathbf{x}, \mathbf{W}), \mathbf{y})}{\partial\mathbf{W}} \tag{3-6}$$

and we hereby simplify $\nabla\mathcal{L}(\mathbf{W}_g^r, \mathcal{B}_k)$ as $\nabla\mathbf{W}$.

The dummy inputs ($\widetilde{\mathbf{x}}$) and labels ($\widetilde{\mathbf{y}}$) are often sampled from a Normal or Uniform distribution denote as $\mathcal{N}(0,1)$ or $\mathcal{U}(0,1)$ respectively. The dummy gradients $\nabla\widetilde{\mathbf{W}}$ are derived similarly as Equation 3-6 by:

$$\nabla\widetilde{\mathbf{W}} = \frac{\partial\mathcal{L}(F(\widetilde{\mathbf{x}}, \mathbf{W}), \widetilde{\mathbf{y}})}{\partial\mathbf{W}} \tag{3-7}$$

The optimization is performed by minimizing a distance D between $\nabla\mathbf{W}$ and $\nabla\widetilde{\mathbf{W}}$ as:

$$(\widetilde{\mathbf{x}}^\star, \widetilde{\mathbf{y}}^\star) = \arg\min_{\widetilde{\mathbf{x}},\widetilde{\mathbf{y}}} D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}). \tag{3-8}$$

Now that the basis of the gradient inversion attack is explained, we will go over the related studies of gradient inversion attacks.

3-5-1 Deep Leakage from Gradients (2019)

The foundational work for gradient inversion attacks "Deep Leakage from Gradients" (DLG) [53] DLG is formalized as optimizing the dummy data by minimizing the Euclidean distance between the original and dummy gradients as such:

$$D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}) = \left\|\nabla\widetilde{\mathbf{W}} - \nabla\mathbf{W}\right\|^2 \tag{3-9}$$

3-5-2 Improved Deep Leakage from Gradients (2020)

Soon after an improvement was made extending DLG (iDLG) [54] by leveraging the classification task and corresponding cross entropy loss to analytically reconstruct the training label $\mathbf{y}$. In a classification task the label $\mathbf{y} = [\mathbf{y}_1, \mathbf{y}_2, \dots]$ is one-hot encoded, meaning that only the ground-truth label $\mathbf{y}_c$ for class c is 1 and the rest of the labels are 0. Furthermore, for a classification task the cross-entropy loss function is used to train the model. This loss function measures how well the model's predicted probabilities match the one-hot encoded ground-truth labels. The gradients of the loss with respect to each output $\mathbf{y}_i$ help us identify the ground-truth label. The gradient g_i of the loss with respect to $\mathbf{y}_i$ is:

$$g_i = \begin{cases} -1 + \frac{\exp\mathbf{y}_i}{\sum_j \exp\mathbf{y}_j} & \text{if } i = c \\ \frac{\exp\mathbf{y}_i}{\sum_j \exp\mathbf{y}_j} & \text{if } i \neq c \end{cases} \tag{3-10}$$

These gradients g_i provide valuable insight because g_i is negative when $i = c$ (the true class) and positive otherwise. This way it is possible to reconstruct the ground-truth labels analytically by taking the signs of the gradients. This technique is however limited to a batch size of one.

3-5-3 Inverting Gradients (2020)

The authors of inverting gradients (InvG) [55] introduce the cosine similarity loss as the distance function for the gradients, as intuitively the dummy data has to move the model parameters in the same hyper-direction as the original data would. Additional to the cosine distance they also include a regularization term which minimizes the total variation for neighbouring pixels [66]. Furthermore, they constrain the search space to images within [0, 1] from prior knowledge about the image modality. The new distance function can be expressed as:

$$D(\nabla \mathbf{W}', \nabla \mathbf{W}) = 1 - \frac{\nabla \widetilde{\mathbf{W}} \cdot \nabla \mathbf{W}}{\left\| \nabla \widetilde{\mathbf{W}} \right\| \left\| \nabla \mathbf{W} \right\|} + \lambda_{\mathrm{TV}} \mathcal{R}_{\mathrm{TV}}(\widetilde{\mathbf{x}}) \tag{3-11}$$

where λ_{TV} is the regularization strength hyperparameter and $\mathcal{R}_{\mathrm{TV}}$ the total variation regularization term:

$$\mathcal{R}_{\mathrm{TV}}(\widetilde{\mathbf{x}}) = \sum_{i,j} \left(\|\widetilde{x}_{i+1,j} - \widetilde{x}_{i,j}\| + \|\widetilde{x}_{i,j+1} - \widetilde{x}_{i,j}\| \right). \tag{3-12}$$

where i indicates the row pixel and j the column pixel.

3-5-4 Self-Adaptive Privacy Attack from Gradients (2020)

An issue with DLG arises when the weights of the network are initialized from a normal distribution instead of a uniform distribution, which causes the attack to not converge properly. To overcome this issue, they propose a weighted Gaussian kernel based function instead of using the Euclidean distance between gradients [56]. The weighted Gaussian kernel based function is defined as:

$$D(\nabla \mathbf{W}', \nabla \mathbf{W}) = Q \cdot \left(1 - \exp\left(\frac{- \left\| \nabla \mathbf{W}' - \nabla \mathbf{W} \right\|^2}{\sigma^2} \right) \right) \tag{3-13}$$

Where Q is a factor for each layer of gradients, weighting layers closer to the input training data more heavily and $\sigma^2 = Var(\nabla \mathbf{W})$.

3-5-5 Recursive Gradient Attack on Privacy (2021)

The Recursive Gradient Attack on Privacy (R-GAP) [57] method is not optimization based, but reconstructs the training data by recursively solving systems of linear equations layer by layer. Unlike optimization-based attacks, R-GAP provides a closed-form solution, making it faster and deterministic, meaning it always produces the same result given the same input. However, it is limited to a batch size of 1, where with larger batch sizes it produces linear combinations of inputs.

3-5-6 See through Gradients (2021)

This method extensively improves the reconstruction capabilities by introducing auxiliary regularization terms and providing a limited method for reconstructing batched labels $\mathbf{y}$. The regularization terms are tailored to images which ensure they are realistic and consistent.

They use the Euclidean distance as in Equation 3-9 and the optimization task is defined as:

$$\widetilde{\mathbf{x}}^{\star} = \arg\min_{\widetilde{\mathbf{x}}} D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}) + \mathcal{R}_{aux}(\widetilde{\mathbf{x}}). \tag{3-14}$$

The labels $\mathbf{y}$ are The batch-wise label restoration only works when each of the ground-truth labels are unqiue. The process starts with batch-wise label restoration, where the algorithm deduces ground truth labels from the signs of gradients at the final classification layer. This helps in initializing the optimization correctly.

For realism, the fidelity regularization term penalizes unrealistic images by using priors such as total variance and ℓ_2 norm penalties, and ensures valid intermediate distributions through batch normalization statistics:

$$\mathcal{R}_{\text{fidelity}}(\widetilde{\mathbf{x}}) = \lambda_{\text{TV}}\mathcal{R}_{\text{TV}}(\widetilde{\mathbf{x}}) + \lambda_{\ell_2}\mathcal{R}_{\ell_2}(\widetilde{\mathbf{x}}) + \lambda_{\text{BN}}\mathcal{R}_{\text{BN}}(\widetilde{\mathbf{x}}). \tag{3-15}$$

To address spatial variance and ensure consistency, group consistency regularization is applied. This works by optimizing multiple versions of the input $\widetilde{\mathbf{x}}$, each initialized with different random seeds. During optimization, it calculates the mean of these multiple versions and penalizes the divergence of each version from this mean, ensuring they converge to a similar solution. The regularization term is:

$$\mathcal{R}_{\text{group}}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{x}}_{g\in\mathcal{G}}) = \lambda_{\text{group}} \left|\widetilde{\mathbf{x}} - \mathbb{E}[\widetilde{\mathbf{x}}_{g\in\mathcal{G}}]\right|_2 . \tag{3-16}$$

where G is the group of randomly initialized inputs and g an individual sample from this group.

3-5-7 Gradient Inversion with Generative Image Prior (2021)

The authors propose a gradient inversion attack using a generative model. The attack reconstructs the input data by navigating through two distinct spaces: the latent space and the parameter space of the generative model. They denote the generative model by $G_\theta : \mathbb{R}^k \to \mathbb{R}^m$, where w represents the parameters of the model, $z \in \mathbb{R}^k$ is a latent code, and $x \in \mathbb{R}^m$ is the generated data.

Latent space search: The latent space $\mathbb{R}^k$ is typically much smaller and more manageable than the full input space $\mathbb{R}^m$. The objective is to find a latent code $\widetilde{\mathbf{z}}$ such that the generated data $G_\theta(\widetilde{\mathbf{z}})$ closely approximates $\mathbf{x}$. This is done by substituting $\widetilde{\mathbf{x}}$ in Equation 3-7 with $G_\theta(\widetilde{\mathbf{z}})$ and optimizing $\widetilde{\mathbf{z}}$ by matching the gradients:

$$\widetilde{\mathbf{z}}^{\star} = \arg\min_{\widetilde{\mathbf{z}}} D(\nabla\widetilde{\mathbf{W}}_{G_\theta(\widetilde{\mathbf{z}})}, \nabla\mathbf{W}). \tag{3-17}$$

Parameter space search: However, since the generative model G_θ may not be perfect, there might still be some error in the reconstruction. After obtaining a good latent representation $\widetilde{\mathbf{z}}$, they propose to fine-tune the model parameters θ in order to better match the gradients:

$$\theta^{\star} = \arg\min_{\theta} D(\nabla\widetilde{\mathbf{W}}_{G_\theta(\widetilde{\mathbf{z}}^{\star})}, \nabla\mathbf{W}). \tag{3-18}$$

3-5-8 Gradient Attack on Transformer-based Language Models (2021)

The gradient inversion attack tailored to transformer-based language models (TAG) focuses specifically Natural Language Processing (NLP) models and on binary classification tasks. NLP task input data involves tokens represented with discrete values. Reconstructing these discrete values requires the attack to optimize in the continuous embedding space and then output the nearest token. The dummy data consists of sequences of dummy input token embeddings $\mathbf{x}$ and dummy labels $\mathbf{y}$.

TAG employs a combination of the Euclidean and Manhattan (L1) distances as its distance function:

$$D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}) = \|\nabla\widetilde{\mathbf{W}} - \nabla\mathbf{W}\|_2 + \lambda_{\text{layer}}(\nabla\mathbf{W})\|\nabla\widetilde{\mathbf{W}} - \nabla\mathbf{W}\|_1 \tag{3-19}$$

where $\lambda_{\text{layer}}(\nabla\mathbf{W})$ is a coefficient parameter based on the layer's proximity to the input data, thereby putting more weight on the differences in gradients of earlier layers. This parameter places more weight on the differences in gradients of earlier layers.

When the dummy embeddings and labels are optimized, the closest discreet token is taken and is mapped with the vocabulary. The attack assumes this vocabulary is also leaked together with the language model or is already known.

3-5-9 Approximate Gradient Inversion Attack (2022)

The Approximate Gradient Inversion Attack (AGIC) [60], specifically targets the Federated Averaging (FedAvg) strategy where multiple local model updates are performed using multiple mini-batches as explained in Section 3-1. The attack implements a efficient one-batch approximation technique, which allows to approximate multiple gradient updates as a single update avoiding costly simulation procedures. This approximation assumes a small (<1e-2) local learning rate and assumes that the aggregated gradient steps caused by the multiple mini-batches can be approximated as a single large batch with gradients:

$$\nabla\mathbf{W} \approx \frac{\mathbf{W}_g^r - \mathbf{W}_k^{r+1}}{-\alpha} \tag{3-20}$$

Furthermore, it employs the same distance function and regularization as Inverting Gradients in Equation 3-11, but it assigns different weights to gradients from different layers to capture their unequal impact on the model's performance, considering factors such as gradient magnitude, layer-specific characteristics, and activation functions like ReLU. The attack also takes advantage of the fact that the same training samples are used across multiple global rounds. By matching updates that contain overlapping samples, AGIC can jointly optimize these updates to enhance the reconstruction quality. This step involves pre-reconstructing images from updates, matching them based on similarity, and then performing joint reconstruction on the matched pairs.

3-5-10 Dropout Inversion Attack (2022)

Dropout is a technique where nodes in a neural network are randomly set to zero during training, preventing overfitting by ensuring that the model does not become overly reliant

on specific nodes. [67] proposed utilizing dropout as a defense mechanism against federated learning attacks. This defense leverages the random nature of dropout to obscure the gradient information, making it more challenging for an attacker to reconstruct the original data.

The Dropout Inversion Attack (DIA) [61] extends the optimization space to account for the probabilistic nature of dropout layers. The attack begins by initializing a dropout mask ψ for each layer l from a Bernoulli distribution given the dropout value p:

$$\Psi = \left\{\psi^{(1)}, \ldots, \psi^{(l)}\right\} \sim \text{Bernoulli}(p) \tag{3-21}$$

These dropout masks are used as deterministic optimizable replacements for the original probabilistic layers. In addition to optimizing dummy inputs and targets, the dropout inversion attack also optimizes the dropout masks of the dropout layers:

$$(\widetilde{\mathbf{x}}^\star, \widetilde{\mathbf{y}}^\star, \Psi^\star) = \underset{\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}}, \Psi}{\arg\min}\, D(\nabla\widetilde{\mathbf{W}}, \nabla\mathbf{W}) \tag{3-22}$$

The objective is to find the combination of input data, labels, and dropout masks that minimize the difference between the gradients of the dummy data and the actual model gradients. This process effectively neutralizes the protective effect of dropout, allowing the attacker to recover the original training data despite the applied dropout defense.

3-5-11 Improved Gradient Inversion Attacks and Defenses (2023)

The authors present an enhanced framework for gradient inversion attacks that addresses both FedSGD and FedAVG strategies [68, 62]. The key components of their approach include zero-shot batch label inference, auxiliary regularizations, multiple updates, and approximation of gradients. The zero-shot batch label inference method is an improvement over those proposed in iDLG [54] or "See through Gradients" [58] because it allows it to handle batches with duplicate labels. Furthermore, to ensure realistic image restoration, the authors introduce auxiliary regularization terms. The auxiliary regularization for fidelity is defined as:

$$\mathcal{R}_{\text{aux}}(\widetilde{\mathbf{x}}) = \lambda_{\text{TV}}\mathcal{R}_{\text{TV}}(\widetilde{\mathbf{x}}) + \lambda_c\mathcal{R}_{\text{clip}}(\widetilde{\mathbf{x}}) + \lambda_s\mathcal{R}_{\text{scale}}(\widetilde{\mathbf{x}}) \tag{3-23}$$

where total variation is set in Equation 3-12, clipping ensures pixel values stay within valid ranges [0, 1], and scaling normalizes pixel values:

$$\mathcal{R}_{\text{clip}}(\widetilde{\mathbf{x}}) = \|\widetilde{\mathbf{x}} - \min(\max(\widetilde{\mathbf{x}}, 0), 1)\|_2 \qquad \mathcal{R}_{\text{scale}}(\widetilde{\mathbf{x}}) = \left\|\widetilde{\mathbf{x}} - \frac{\widetilde{\mathbf{x}} - \min_v}{\max_v - \min_v}\right\|_2.$$

These regularization terms keep the images more bounded between 0 and 1. Furthermore, the authors propose to improve reconstruction quality by leveraging multiple model updates. They make the assumption that the attacker can gather multiple gradient updates caused by the same mini-batch and use these multiple gradients for more detailed and accurate reconstructions. For FedAvg, they make the approximation, similar to AGIC [60],

$$\nabla\mathbf{W} \approx \frac{W_0 - W_E}{-E\eta} \tag{3-24}$$

with E local training steps and local learning rate η.

3-5-12 Learning to Invert (2023)

The "Learning to Invert" (LTI) [63] method presents a novel approach to gradient inversion attacks by leveraging a model trained on an auxiliary dataset to recover client samples from gradients. This method circumvents traditional defenses, such as differential privacy and gradient compression, which were previously effective against optimization-based attacks.

The LTI method operates under the assumption that the attacker has access to an auxiliary dataset that shares a similar distribution with the private data. The auxiliary dataset is used to train a gradient inversion model g_θ that predicts the batch of data points $\mathcal{B}_k$ from the gradient $\nabla \mathbf{W}$. The training objective is to minimize the reconstruction error on the auxiliary dataset:

$$\min_\theta \mathbb{E}_{\mathcal{B}_k \sim \mathcal{D}_{\text{aux}}} \mathcal{L}_{\text{attack}}(g_\theta(\nabla \mathbf{W}), \mathcal{B}_k) \tag{3-25}$$

Here, $\mathcal{L}_{\text{attack}}$ is the loss function designed to be permutation invariant with respect to the batch $\mathcal{B}_k$ because different permutations of samples can result in the same gradients. This is achieved by defining $\mathcal{L}_{\text{attack}}$ as [69]:

$$\mathcal{L}_{\text{attack}}(g_\theta(\nabla \mathbf{W}), \mathcal{B}_k) = \min_\pi \sum_{i=1}^{B} \mathcal{L}_{\text{attack}}^{\text{single}}(\nabla \mathbf{W}_i, (\mathbf{x}_{\pi(i)}^{\text{aux}}, \mathbf{y}_{\pi(i)}^{\text{aux}})) \tag{3-26}$$

where π is a permutation of the batch elements, and $\mathcal{L}_{\text{attack}}^{\text{single}}$ is the loss used to train the federated global model and $(\mathbf{x}_{\pi(i)}^{\text{aux}}, \mathbf{y}_{\pi(i)}^{\text{aux}})$ are input and output pairs from the auxilary dataset. LTI is different from optimization-based methods by not explicitly incorporating data priors into the objective function. Instead, it learns the data properties from the auxiliary dataset. This allows LTI to adapt to different defense mechanisms without the need for carefully designed objective functions tailored to each defense.

3-5-13 Tabular Data Leakage (2023)

The TabLeak [65] method addresses the unique challenges of reconstructing tabular data in federated learning, which involves both discrete and continuous features. For discrete features, typically one-hot encoded, TabLeak employs a softmax relaxation to transform the optimization problem into a continuous one. Specifically, for a discrete feature vector z with k categories, the softmax function, denoted by σ, ensures the outputs are within [0, 1] and sum to one, maintaining the one-hot encoding constraint:

$$\sigma(z)[j] = \frac{\exp(z[j])}{\sum_{i=1}^{k} \exp(z[i])} \quad \forall j \in \{1, \ldots, k\} \tag{3-27}$$

This approach allows gradient-based optimization methods like Adam to handle discrete data by working in the continuous domain, while still approximating the discrete structures.

To enhance the robustness of the reconstructions, TabLeak uses pooled ensembling. This involves running multiple independent optimization processes with different initializations and then aggregating the results by taking the median values of the reconstructed features.

This method effectively reduces variance and improves the accuracy of the final reconstruction by leveraging the consistency across multiple reconstructions.

Finally, to quantify the reliability of the reconstructed features, TabLeak introduces ensemble confidence. By measuring the agreement among the multiple reconstructions, features with low variance (high agreement) are considered more reliable. This systematic evaluation provides a confidence score for each feature, crucial for assessing the quality of reconstructions in the tabular domain where direct human inspection is challenging.

3-5-14 Discussion

The exploration of gradient inversion attacks reveals a rich and evolving landscape of techniques aimed at reconstructing training data from gradients. The foundational work "Deep Leakage from Gradients" (DLG) [53] set the stage by demonstrating the feasibility of these attacks using Euclidean distance minimization. The subsequent enhancement, "Improved Deep Leakage from Gradients" (iDLG) [54], advanced this by analytically reconstructing training labels, though it remained limited to a batch size of one.

"Inverting Gradients" (InvG) [55] introduced cosine similarity and total variation regularization, improving the attack's robustness, particularly for image data. The "Self-Adaptive Privacy Attack from Gradients" (SAPAG) [56] tackled convergence issues arising from different weight initializations, further refining the optimization process with a Gaussian kernel-based distance function.

The "Recursive Gradient Attack on Privacy" (R-GAP) [57] deviated from optimization-based methods, offering a faster, deterministic solution by recursively solving linear equations, albeit limited to single-sample batches. "See through Gradients" [58] and "Gradient Inversion with Generative Image Prior" [59] provided significant advancements in reconstruction quality and scalability to larger batches, leveraging auxiliary regularizations and generative models.

The "Gradient Attack on Transformer-based Language Models" (TAG) [64] addressed the unique challenges of NLP models, using combined Euclidean and Manhattan distances for gradient comparison. The "Approximate Gradient Inversion Attack" (AGIC) [60] efficiently approximated multiple gradient updates in federated averaging, enhancing practicality for real-world federated learning setups.

The "Dropout Inversion Attack" (DIA) [61] extended the attack surface by accounting for dropout layers, effectively neutralizing this common defense mechanism. "Improved Gradient Inversion Attacks and Defenses" [62] proposed a comprehensive framework addressing both FedSGD and FedAvg strategies, introducing zero-shot batch label inference and auxiliary regularizations for realistic image restoration.

"Learning to Invert" (LTI) [63] presented a paradigm shift by using a model trained on an auxiliary dataset to predict client samples from gradients, overcoming traditional defenses like differential privacy. "Tabular Data Leakage" (TabLeak) [65] innovatively tackled the complexities of reconstructing tabular data, employing softmax relaxation and pooled ensembling for improved accuracy and reliability.

3-5-15 Research Gap

Despite the significant advancements in gradient inversion attacks, several gaps remain. Notably, none of the existing studies address time series forecasting tasks. Time series forecasting is a regression task optimized with loss functions such as Mean Squared Error (MSE), whereas existing studies focus on classification tasks using cross-entropy loss. The analytical reconstruction of labels, feasible with cross-entropy loss in classification tasks, has not been explored for regression tasks trained with MSE. Furthermore, forecasting targets are often continuous values, contrasting with the discrete, one-hot encoded labels in classification.

Additionally, none of the attacks investigate model architectures specifically designed for temporal data, such as Temporal Convolutional Networks or Gated Recurrent Units. Given the increasing literature on federated learning for load forecasting, this gap highlights a critical area for further research. Our study aims to bridge this gap by extending gradient inversion attacks to time series forecasting tasks, exploring the unique challenges and implications for federated learning in this domain. This research will contribute to a deeper understanding of gradient inversion attacks and inform the development of more robust security and privacy measures for federated learning applications involving temporal data.

Chapter 4

Additional Experiments

This chapter provides more details about the experimental setup in Section 4-1. We describe the datasets used in the paper and the experiments in Section 4-2. Afterwards, in Section 4-3, we present additional baseline attack experiments and results. Finally, in Section 4-4, we present additional experiments and results regarding the different gradient distance functions, the gradient inversion model quantile predictions, and regularization techniques.

4-1 Experimental Setup Details

In this section we go over the datasets and how they are gathered, the data normalization technique used to bound all data between 0 and 1. Furthermore, we give a in depth description of the models used in the paper, and the gradient inversion model architecture and hyperparameters. Moreover, we explain the used evaluation metric.

4-1-1 Datasets

Each dataset contains columns and rows, where the columns represent a time series and a row a timestamp. For the proprietary, london smartmeter, and KDDCup 2018 we pick the first column as the time series to be attacked. For Electricity 370 we picked the fourth column, because the first three had large changes in their characteristics throughout the sequence which would have required more pre-processing. Each of these time series are explore in Section 4-2.

Proprietary Dataset: The dataset contains active power (kW) data of 37 households ranging from 2021-01-01 to 2023-01-01 sampled every 15 minutes.

London Smartmeter Dataset: The dataset contains power consumption of 5560 households ranging from November 2011 to February 2014 sampled every 30 minutes. The time series represent the energy consumption in kilowatt hour (kWh).[1] In the experiments we take the 1-st column as data silo sequence.

[1] https://zenodo.org/records/4656091

Electricity 370 Dataset: The Dataset contains kW measured every 15 minutes of 370 'clients'. The data was gathered between 2011 and 2014, however, not all clients' measurements start at 2011, some clients were introduced after.[2] In the experiments we take the 4-th column as data silo sequence.

KDDCup 2018 (Air Quality) Dataset: The dataset contain hourly sampled time series which represents the air quality levels in 59 locations in 2 cities. 35 stations in Beijing and 24 stations in London.[3] In the experiments we take the 1-st column as data silo sequence.

4-1-2 Data Normalization

In the paper we normalize all of the time series using the min-max normalization technique. It scales the values of a dataset to a specific range, in our case [0, 1], which helps in improving the performance and convergence of deep learning algorithms. This scaling method is beneficial because it ensures that all features contribute more equally to the result and prevents features with larger ranges from dominating the learning process.

The formula for min-max normalization for a given data point s_t in a time series is:

$$s'_t = \frac{s_t - \min(s)}{\max(s) - \min(s)} \tag{4-1}$$

where s_t is the original value at time t, s'_t is the normalized value at time t, $\min(s)$ is the minimum value in the time series and $\max(s)$ is the maximum value in the time series.

This formula linearly transforms the original data into a scaled version where the minimum value of the original data maps to 0 and the maximum value maps to 1. This transformation preserves the relationships between the original data points while constraining them to a common range. This facilitates a more uniform input to various algorithms, particularly those sensitive to the scale of input data, like neural networks.

4-1-3 Model Architectures

This section describes the model architectures that are attacked and evaluated in the experiments. For each predictor, the input is the observation sequence $\mathbf{S}_{\text{obs}} \in \mathbb{R}^{H \times d}$, as described in Section 3-2. The predictions are $\hat{\mathbf{s}}_{\text{tar}} \in \mathbb{R}^{F \times 1}$, where F is the desired output sequence length.

FCN Predictor: The model is a fully connected neural network where the input is flattened into a vector and passed through two linear hidden layers with size h. Each hidden layer is followed by a sigmoid activation function. The final linear layer maps the hidden representation to the output sequence

CNN Predictor: This model architecture is based on that of the LeNet architecture proposed in "Improved Deep Leakage of Gradients" [54]. The model is a convolutional neural network and the input is passed through three convolutional layers, each applying a 1D convolution followed by a sigmoid activation function. The sequence of convolutions is defined as follows:

[2] https://archive.ics.uci.edu/dataset/321/electricityloaddiagrams20112014
[3] https://zenodo.org/records/4656756

The first convolutional layer has d input channels, hidden size h output channels, a kernel size of 5, padding of 2, and a stride of 2. The second convolutional layer has hidden size h input channels, h output channels, a kernel size of 5, padding of 2, and a stride of 2. The third convolutional layer has hidden size h input channels, h output channels, a kernel size of 5, padding of 2, and a stride of 1.

After the convolutional layers, the output is flattened into a one-dimensional vector. This flattened vector is then passed through a fully connected layer that outputs the prediction.

TCN Predictor: The model contains a temporal convolutional network (TCN), as implemented by [24], where the input is processed through several levels of causal convolutional layers, calculated to encapsulate the entire input sequence length, with each layer using a kernel size of 6 and a dilation factor of 2. Dropout is applied with a probability of 0.1, and the activation function used is ReLU. Weight normalization is also enabled to improve training stability. After passing through the TCN layers, the last hidden state is mapped to the final prediction by a fully connected layer.

GRU-2-FCN Predictor: The model is a gated recurrent unit (GRU) network where the input is processed through a encoder GRU with a hidden size of h, capturing the temporal dependencies in the sequence. The GRU layer outputs a hidden state at each time step, and the final hidden state is passed through a fully connected layer that maps the hidden representation to the predictions.

GRU-2-GRU Predictor: This model architecture is based on the Sequence-2-Sequence model [70]. The input is processed similarly to the GRU-2-FCN architecture, however, the final hidden state of the encoder GRU is used as the initial hidden state for the decoder.

The decoder is another GRU layer that takes the final hidden state from the encoder and processes it to generate the output sequence. The decoder GRU outputs the hidden state at each time step, which is passed through a fully connected layer that maps the hidden representation to an output value per timestep.

4-1-4 Gradient Inversion Model

Model Architecture: The gradient inversion model is a custom architecture with two modules, one for outputting quantiles for observations and one for targets given gradients $\nabla \mathbf{W}$. Each module contains two residual blocks with hidden sizes of 768 and 512, respectively. Each residual block consists of fully connected layers, ReLU activations, batch normalization, and dropout for regularization, with an optional adaptation layer ensuring matching dimensions for identity mapping. Additionally, each module includes a fully connected layer that outputs the quantiles for its observation or target reconstruction.

Training Hyperparameters: The gradient inversion model predicting quantiles is trained for 75 epochs with a learning rate of 1e-3 and a batch size of 32. The hidden sizes of the residual layers are 768 and 512 respectively. The set quantiles are $[0.1, 0.3, 0.7, 0.9]$ and is optimized with the AdamW optimizer. There is a learning rate scheduler based on the gradient matching loss, that reduces the learning rate 10x, when it does not decrease for at least 500 steps.

4-1-5 Evaluation Metric

All of the numeric results in the paper and here in this section are calculated using the Symmetric Mean Absolute Percentage Error (sMAPE) [71]. It is a normalized error metric that ranges between 0 and 2, allowing for easy comparison between different datasets. This normalization is achieved by scaling the absolute difference between the actual values ($\mathbf{S}$) and the predicted values ($\hat{\mathbf{S}}$) by the sum of their absolute values. The formula for sMAPE is given by:

$$\text{sMAPE} = \frac{1}{N} \sum_{i=1}^{N} \frac{2 \left| \mathbf{S}_i - \hat{\mathbf{S}}_i \right|}{|\mathbf{S}_i| + \left| \hat{\mathbf{S}}_i \right|} \tag{4-2}$$

where N are the number of elements in sequence $\mathbf{S}$, and $|\cdot|$ is the absolute value function.

This normalization ensures that the error measure is bounded, facilitating straightforward comparison across datasets with different scales and units. The closer the sMAPE value is to 0, the better, as it indicates higher similarity between the actual and predicted data.

4-2 Dataset Exploration

In this section, we explore the datasets used in our analysis, examining different resolution levels to uncover their time series characteristics as explained in Section 3-2. The exploration includes entire sequences, a single month, weekly profiles, and an individual sample, providing a detailed view of the data. These visualizations help identify seasonal trends, cycles, and other possible characteristics or insights. We analyze four datasets: Electricity 370, KDD Cup 2018, London Smartmeter, and a Proprietary dataset. Each dataset is presented through figures of resolutions and if they are a electricity dataset, the weekly profiles are also shown.

4-2-1 Electricity 370

Figure 4-1 encapsulates nearly two years of data, revealing a clear seasonal trend. The load fluctuates significantly over the year, with higher consumption during winter months and a noticeable dip in summer. This seasonality likely indicates a higher demand for heating in colder months, which is a reasonable assumption for many regions.

Figure 4-2 zooms into a single month timeframe, displaying daily patterns. The electricity usage exhibits a pronounced diurnal cycle with peaks during the day and dips at night. This daily periodicity aligns with human activity patterns, where daytime usage is higher due to work and household activities, and nighttime usage drops as people sleep.

Figure 4-3 and Figure 4-4 further dissect the data by days of the week. Both profiles show consistent peaks and dips across the week, but with some variations. Mondays exhibit a high load, likely due to the resumption of activities after the weekend, while weekends show relatively lower and more consistent usage patterns. The median load profile tends to smooth out extreme values, providing a clearer view of typical daily consumption without the influence of outliers.

Finally, Figure 4-5 presents a single sample as an example, with one day of observations and one day as the target to be forecasted.

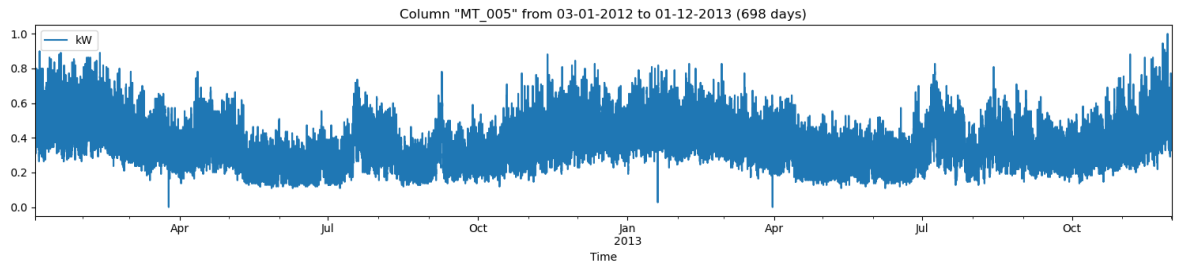


Figure 4-1: Electricity 370: Whole Sequence of Column 5

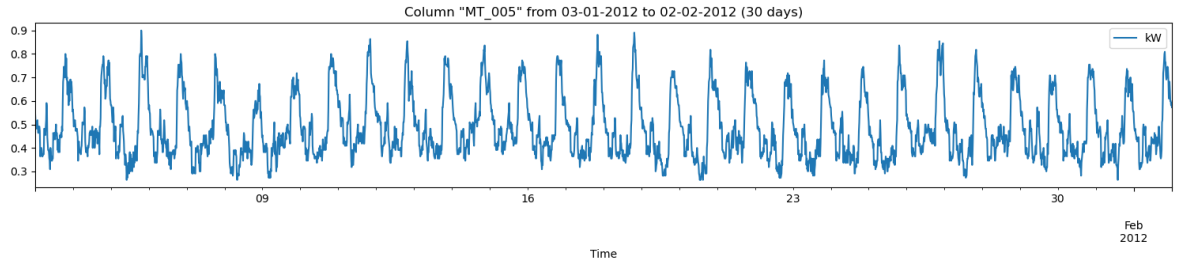


Figure 4-2: Electricity 370: First Month of column 5 Sequence

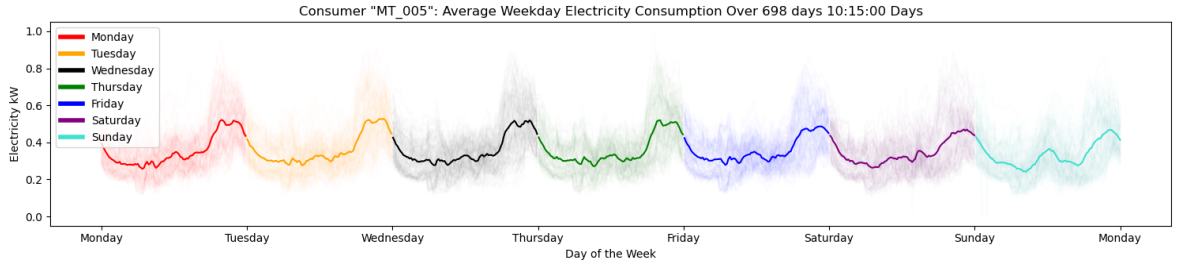


Figure 4-3: Electricity 370: Column 5 Average Load Profile

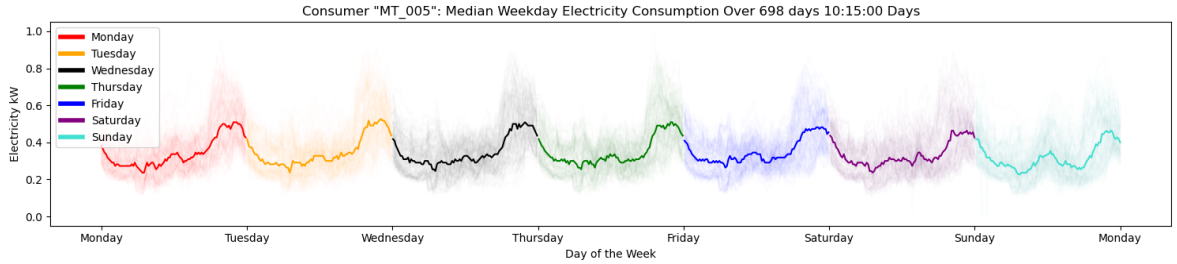


Figure 4-4: Electricity 370: Column 5 Median Load Profile

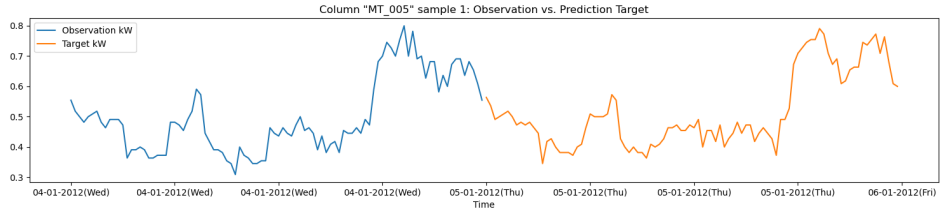


Figure 4-5: Electricity 370: The First Sample of Column 5

4-2-2 KDDCup 2018

PM2.5 refers to particulate matter with a diameter of less than 2.5 micrometers, which can penetrate the lungs and enter the bloodstream, making it a significant health concern. The figures provided show the PM2.5 time series data from the KDD Cup 2018 dataset, focusing on air quality in Beijing. Figure 4-6 displays the entire sequence from January 7, 2017, to October 18, 2017. While there are no clear seasonal patterns, the PM2.5 levels tend to be lower during the summer months of June and July. Throughout the year, there are several sudden increases and decreases, indicating occasional pollution events that could be due to various environmental or human activities.

Looking closer, Figure 4-7 shows the first month of data, revealing a weekly pattern, although the intervals are not regular. Significant spikes, especially towards the end of January, suggest short-term pollution episodes. Figure 4-8 provides a sample of five days of observations followed by a one-day target period, highlighting the variability in PM2.5 levels over a short time. This sample shows the unpredictable nature of air quality, emphasizing the challenges in forecasting. Together, these figures offer a detailed view of Beijing's PM2.5 levels, highlighting both long-term trends and short-term changes.

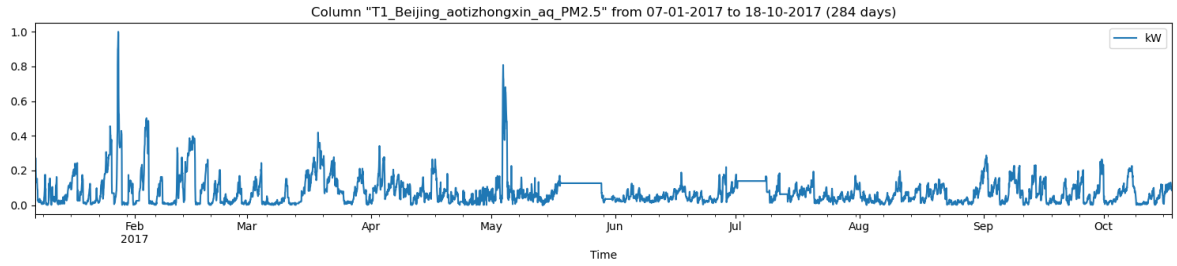


Figure 4-6: KDD Cup: Whole Sequence of Column 1

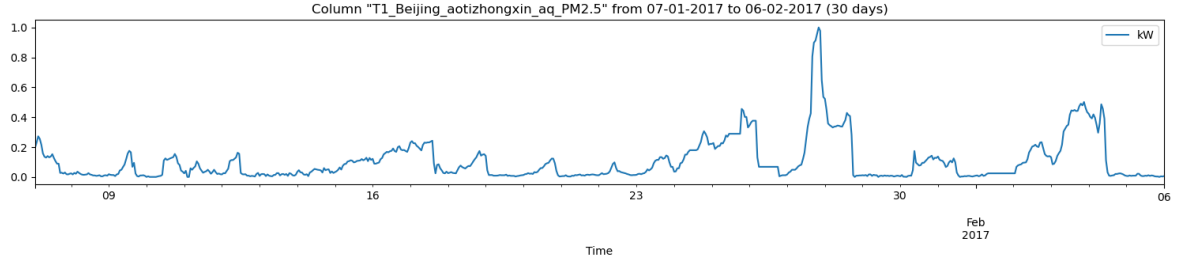


Figure 4-7: KDD Cup: First Month of Column 1 Sequence

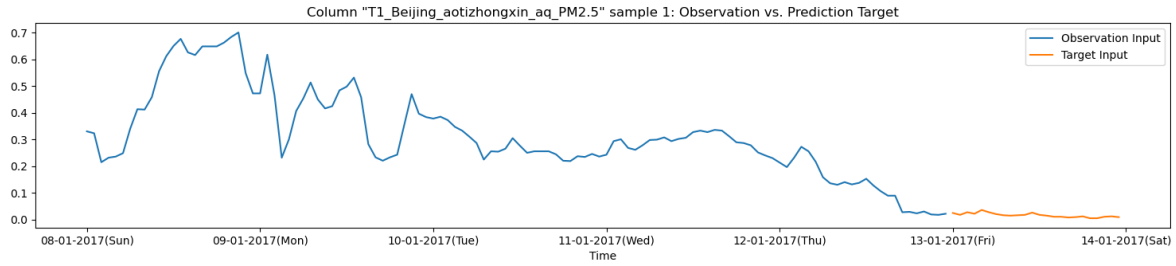


Figure 4-8: KDD Cup: The First Sample of Column 1

4-2-3 London Smartmeter

Figure 4-9 encapsulates nearly a year of data, revealing various trends and anomalies in the household's electricity consumption. Notably, there is a significant increase in consumption towards the end of February and throughout March. While there isn't a clear distinction between colder and warmer months in terms of overall load, there are several sudden drops in energy usage. However, these drops do not reach zero, indicating some minimal level of consumption, possibly due to idling devices, remains constant.

Figure 4-10 zooms into a single month's timeframe, showcasing clear daily cycles of day and night. The electricity usage exhibits pronounced diurnal patterns, with distinct peaks during the day and troughs at night, aligning with typical household activities. Figure 4-11 and Figure 4-12 further analyses the data, highlighting consistent daily cycles but showing no significant differences between weekends and workdays. Both profiles emphasize regular peaks during active hours and troughs during idle periods.

Finally, Figure 4-13 presents a single sample as an example, displaying one day of observations and one day as the target for forecasting. This figure highlights abrupt spikes in energy consumption, reflecting short-term fluctuations that pose a challenge for accurate prediction. These spikes suggest sporadic high-energy activities, adding complexity to modeling and forecasting future consumption patterns based on past data.

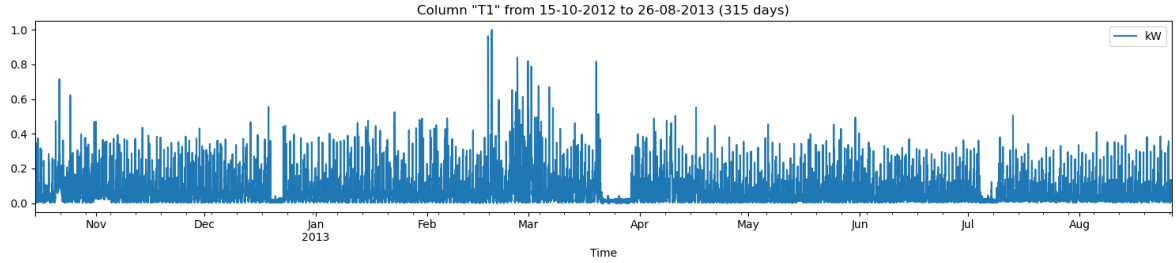


Figure 4-9: London Smartmeter: Whole Sequence of Column 1

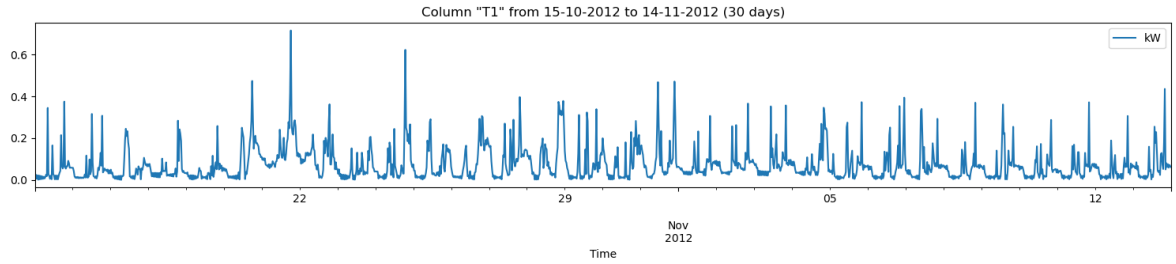


Figure 4-10: London Smartmeter: First Month of Column 1 Sequence

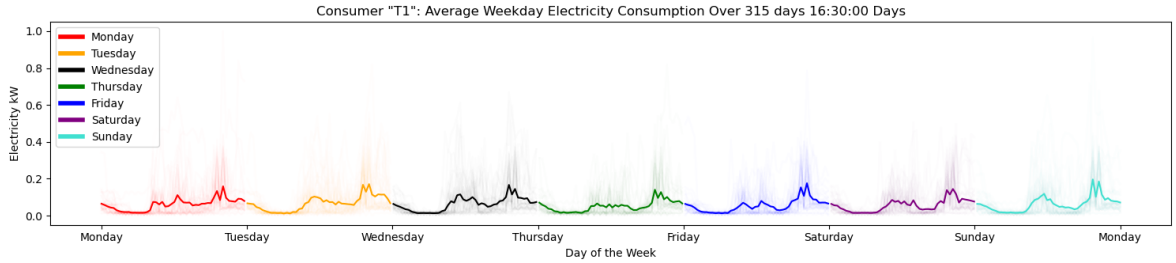


Figure 4-11: London Smartmeter: Column 1 Average Load Profile

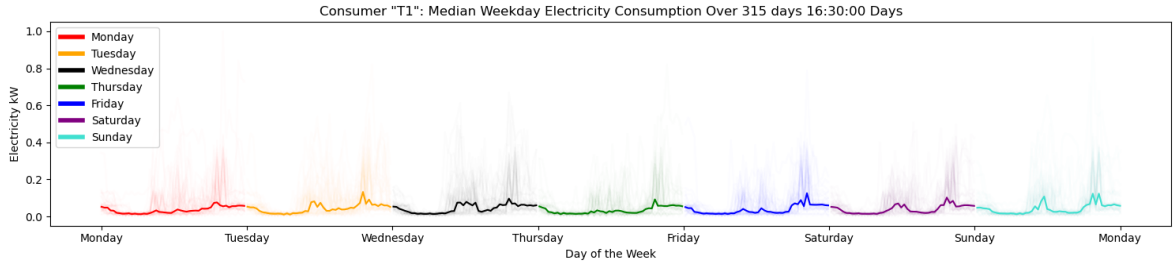


Figure 4-12: London Smartmeter: Column 1 Median Load Profile

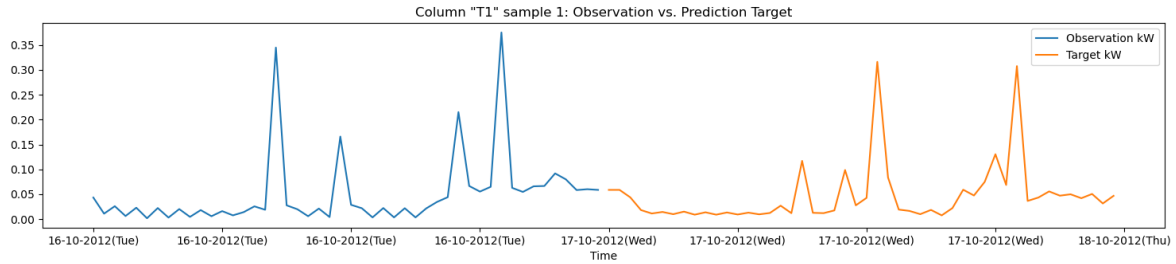


Figure 4-13: London Smartmeter: The First Sample of Column 1

4-2-4 Proprietary Data

Examining the entire sequence of electricity consumption over 464 days, as depicted in Figure 4-14, does not reveal clear seasonal fluctuations. There is no distinct difference between summer and winter months. However, a noticeable sudden drop in December where the values drop to 0.0 indicates a possible outage of the sensor. Focusing on the first month of the sequence in Figure 4-15 provides a more granular view, where the periodicity of daily patterns becomes evident. Notable peaks correspond to higher usage during waking hours and reduced consumption during nighttime, emphasizing the regular day-night cycle.

The average load profile in Figure 4-16 and the median load profile in Figure 4-17 illustrate typical daily consumption patterns over the entire dataset. Both figures indicate an increase in energy use during weekends, likely due to the activities of a working household. During weekdays, consumption peaks in the morning and evening hours, reflecting common household activities. Additionally, Figure 4-18 presents a specific observation and target day, highlighting consistent energy usage with sudden increases and decreases, which reflect the variability of household activities.

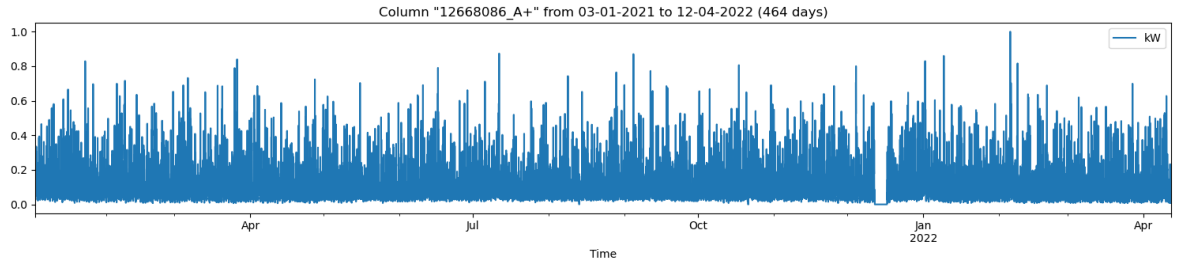


Figure 4-14: Proprietary: Whole Sequence of Column 1

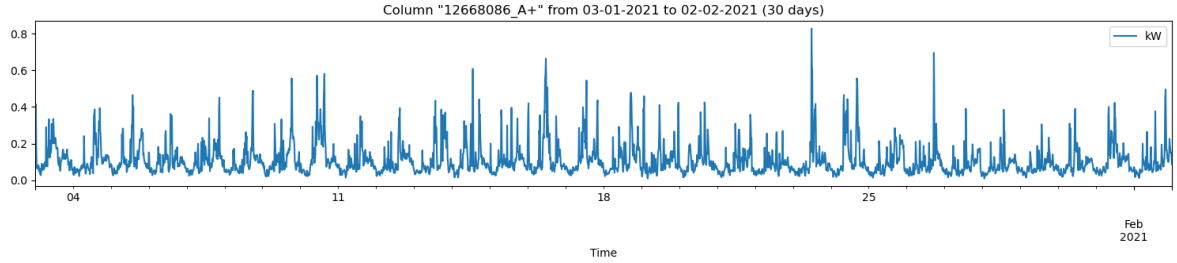


Figure 4-15: Proprietary: First Month of Column 1 Sequence

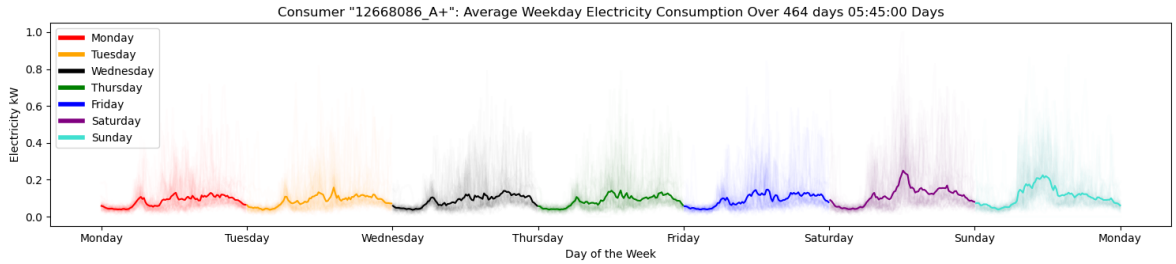


Figure 4-16: Proprietary: Column 1 Average Load Profile

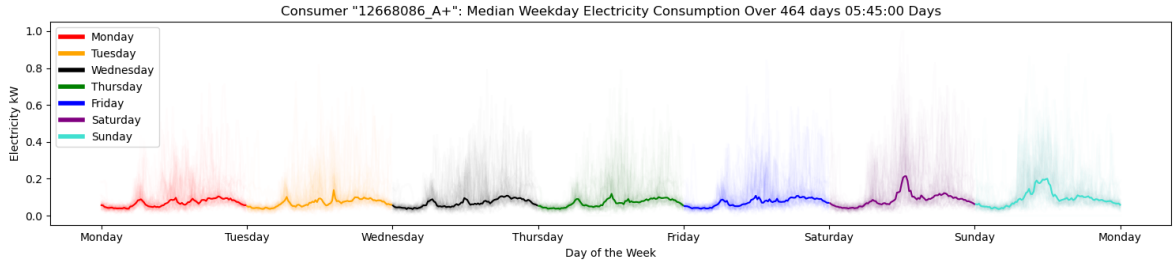


Figure 4-17: Proprietary: Column 1 Median Load Profile

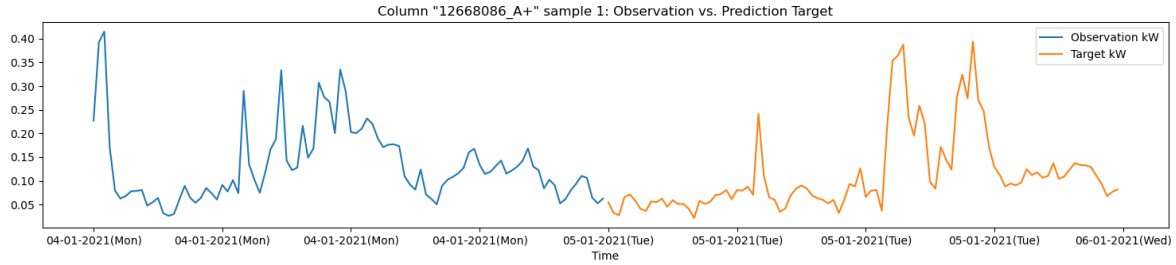


Figure 4-18: Proprietary: The First Sample of Column 1

4-3 Baseline Attacks

As described in the paper, we evaluate the Deep Leakage of Gradients (DLG) [53] with the LBFGS and Adam optimizers, Inverting Gradients (InvG) [55], Dropout Inversion Attack (DIA) [61] and Learning To Invert (LTI) [63] baselines. Their methodologies are further explained in Section 3-5. Their configurations are taken from their open-sourced repositories: DLG-LBFGS[4], DLG-Adam[5], DIA[6], and LTI[7]. The repository of LTI had no implementation for reconstructing both the observations and targets.

4-3-1 Baseline Reconstructions on Datasets

Figure 3 in the paper only shows the reconstructions on a single dataset. In Figure 4-19 these results are also shown for the other 3 datasets.

DLG-LBFGS: Although this attack converged quickly it is rather limited in its ability to reconstruct the time series data. It is only able to reconstruct the observations for the FCN architecture. It is able to reconstruct the targets for most datasets and architectures. However, the TCN architecture is too difficult because of the dropout layers.

DLG-Adam: The more sophisticated optimizer improves its ability to reconstruct both observations and targets. It is even better at reconstructing targets than the Cosine Similarity based attacks InvG and DIA. Also, it is able to reconstruct pats of the observations and targets for the TCN, although they are noisy.

InvG: These reconstructions back up the claim in the paper that the Cosine Similarity loss is unable to reconstruct the targets completely. In the Electricity 370 dataset, some targets seem to have the same shape, but are offset down a bit, indicating that this last bit is due to some lacking detail in the gradient matching.

DIA: Even though this attack also optimizes the dropout masks, to counter its defensive properties as seen in the reconstructions of previous attacks, it is unable to properly reconstruct the whole observation or target sequence for the TCN architecture.

LTI: This attack achieves less fine grained reconstructions and in some cases is unable to reconstruct the fine details of the time series.

[4] https://github.com/mit-han-lab/dlg
[5] https://github.com/JonasGeiping/breaching
[6] https://github.com/dAI-SY-Group/DropoutInversionAttack
[7] https://github.com/wrh14/Learning_to_Invert

Figure 4-19: Grid reconstructions for all baselines with attack method X on dataset X using models FCN, CNN, GRU, and TCN.

4-3-2 Baseline Attacks on GRU-based Architectures

As described in the paper, GRUs selectively remember and forget information through their reset and update gates [23]. Furthermore, the gradients of the GRU do not directly relate to individual inputs, instead they are averaged over all inputs because of the sequential nature of Recurrent Neural Networks. This many-to-one mapping makes it challenging to reconstruct the inputs. For this we have conducted additional experiments with the baselines. In Figure 4-20 the results are shown of using different baselines attacking the GRU-based architectures. These hypotheses are supported by the results of the optimization-based baselines (DLG-Adam and InvG). The DLG-Adam attack, using the Euclidean distance, is able to reconstruct the targets, but the observations also lack detail. The InvG attack, using the Cosine Similarity loss is actually optimizing the inputs outside of the domain $[0, 1]$. Especially the Learning to Invert (LTI) [63] seemed promising because it learns the relation between gradients and inputs, potentially learning this non-linear relation. Showing some similarity in both the GRU-2-FCN and GRU-2-GRU architectures. However, this seems to be the case only for the Electricity 370 dataset. For the Proprietary dataset none of the details remain.

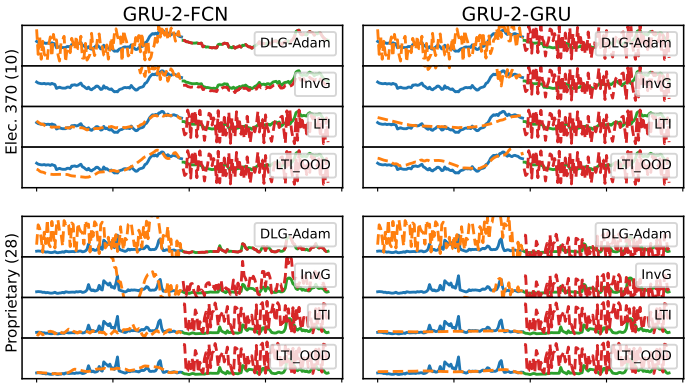


Figure 4-20: Grid reconstructions for attacking Seq2Seq model on 24-05-2024.

4-3-3 Total Variation Effects on Reconstruction

The authors of Inverting Gradients [55] propose the additional regularization of total variation in Equation 3-12 which reduces the difference between neighbouring pixels. This technique can also be applied on time series, to reduce the difference between neighbouring values. However, as described in the paper this has a negative effect on the reconstruction accuracy. In the paper we present a table that illustrates this reduce in performance. Here in this section we present the corresponding reconstruction plots in Figure 4-21. The observation reconstructions decrease in accuracy when the $\lambda_{\text{TV}}^{tar}$ increases. Furthermore, the when increasing the regularization term for the observation $\lambda_{\text{TV}}^{obs}$, it can be seen that it decreases the fine detail present in the Proprietary dataset.

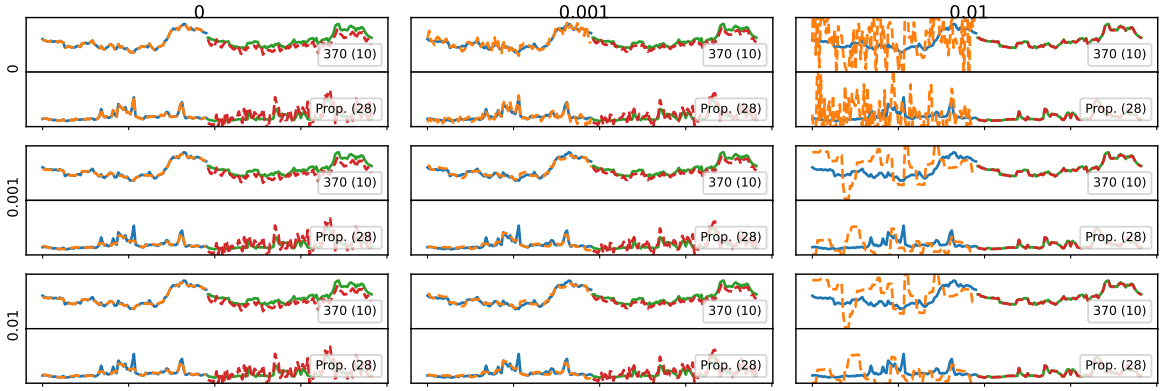


Figure 4-21: Reconstructions of Inverting Gradients with different hyperparameters for total variation on the observation sequence $\lambda_{\text{TV}}^{obs}$ (rows) and the target sequence $\lambda_{\text{TV}}^{tar}$ (columns). The reconstructions are of the Electricity 370 (370 in the figure) and Proprietary (Prop. in figure) datasets.

4-4 TS-Inverse Experiments

4-4-1 Gradient Distance Functions

In the paper the gradient distances are only evaluated against the TCN architecture, because as the following results show, for the CNN and FCN models there is no consistent improvements among the distance functions.

In Table 4-1 and Table 4-2, the Cosine, Cosine + L1-Norm, Cosine + L2-Norm, L2-Norm, and L1-Norm gradient distance functions are evaluated on the FCN, CNN, and TCN architectures.

For the CNN architecture, it is evident that for batch sizes 2 and 4 in the Electricity 370 dataset (Table 4-1), none of the loss functions show consistent improvements in reconstructing the observations or targets. In the London Smartmeter dataset (Table 4-2), there are larger improvements, but not consistently with a single distance function.

For the FCN architecture on the Electricity 370 dataset (Table 4-1), with batch sizes 2 and 4, the Cosine + L2-Norm performs best for reconstructing both the observations and targets. On the London Smartmeter dataset (Table 4-2), there are no significant improvements either.

For the TCN architecture, as presented in the paper, the distance functions involving the L1-Norm consistently achieve the best results.

	Model	CNN		FCN		TCN	
Batch Size	Gradient Loss	Observation	Target	Observation	Target	Observation	Target
1	Cosine + L1-Norm	$0.002_{0.00}$	$\mathbf{1.4e\text{-}07}_{0.00}$	$3.3e\text{-}07_{0.00}$	$\mathbf{7.4e\text{-}08}_{0.00}$	$0.083_{0.01}$	$\mathbf{9.3e\text{-}08}_{0.00}$
	Cosine + L2-Norm	$\mathbf{8.9e\text{-}04}_{0.00}$	$\mathbf{1.4e\text{-}07}_{0.00}$	$2.8e\text{-}07_{0.00}$	$\mathbf{7.4e\text{-}08}_{0.00}$	$0.290_{0.17}$	$\mathbf{9.3e\text{-}08}_{0.00}$
	Cosine	$0.005_{0.00}$	$\mathbf{1.4e\text{-}07}_{0.00}$	$0.004_{0.00}$	$\mathbf{7.4e\text{-}08}_{0.00}$	$0.377_{0.19}$	$\mathbf{9.3e\text{-}08}_{0.00}$
	Euclidean	$0.002_{0.00}$	$\mathbf{1.4e\text{-}07}_{0.00}$	$7.7e\text{-}07_{0.00}$	$\mathbf{7.4e\text{-}08}_{0.00}$	$0.236_{0.05}$	$\mathbf{9.3e\text{-}08}_{0.00}$
	L1	$0.002_{0.00}$	$\mathbf{1.4e\text{-}07}_{0.00}$	$\mathbf{2.0e\text{-}07}_{0.00}$	$\mathbf{7.4e\text{-}08}_{0.00}$	$\mathbf{0.081}_{0.03}$	$\mathbf{9.3e\text{-}08}_{0.00}$
2	Cosine + L1-Norm	$0.167_{0.09}$	$0.464_{0.10}$	$0.150_{0.06}$	$0.433_{0.08}$	$\mathbf{0.362}_{0.14}$	$\mathbf{0.053}_{0.04}$
	Cosine + L2-Norm	$0.170_{0.09}$	$0.464_{0.10}$	$\mathbf{0.136}_{0.04}$	$\mathbf{0.368}_{0.02}$	$0.628_{0.24}$	$0.119_{0.03}$
	Cosine	$0.170_{0.09}$	$\mathbf{0.455}_{0.09}$	$0.167_{0.08}$	$0.462_{0.11}$	$0.619_{0.18}$	$0.630_{0.40}$
	Euclidean	$0.169_{0.09}$	$0.467_{0.10}$	$0.160_{0.08}$	$0.484_{0.09}$	$0.629_{0.32}$	$0.613_{0.40}$
	L1	$\mathbf{0.167}_{0.09}$	$0.464_{0.10}$	$0.151_{0.06}$	$0.436_{0.09}$	$0.471_{0.08}$	$0.095_{0.06}$
4	Cosine + L1-Norm	$0.206_{0.06}$	$0.595_{0.10}$	$0.266_{0.10}$	$0.557_{0.08}$	$0.647_{0.11}$	$0.261_{0.17}$
	Cosine + L2-Norm	$0.207_{0.06}$	$0.593_{0.10}$	$\mathbf{0.233}_{0.11}$	$\mathbf{0.515}_{0.07}$	$0.715_{0.16}$	$0.901_{0.20}$
	Cosine	$0.207_{0.06}$	$\mathbf{0.553}_{0.06}$	$0.271_{0.12}$	$0.580_{0.04}$	$0.738_{0.18}$	$0.818_{0.33}$
	Euclidean	$\mathbf{0.205}_{0.06}$	$0.598_{0.10}$	$0.253_{0.09}$	$0.577_{0.09}$	$0.742_{0.16}$	$0.811_{0.31}$
	L1	$0.206_{0.06}$	$0.594_{0.10}$	$0.264_{0.10}$	$0.557_{0.08}$	$\mathbf{0.554}_{0.11}$	$\mathbf{0.131}_{0.10}$

Table 4-1: Gradient distance function comparison on Electricity 370 (seeds 10, 43, 28)

	Model	CNN		FCN		TCN	
Batch Size	Gradient Loss	Observation	Target	Observation	Target	Observation	Target
1	Cosine + L1-Norm	$0.029_{0.01}$	$\mathbf{2.3e\text{-}06}_{0.00}$	$\mathbf{1.3e\text{-}05}_{0.00}$	$\mathbf{1.6e\text{-}06}_{0.00}$	$0.212_{0.03}$	$\mathbf{2.3e\text{-}06}_{0.00}$
	Cosine + L2-Norm	$\mathbf{0.005}_{0.00}$	$\mathbf{2.3e\text{-}06}_{0.00}$	$2.1e\text{-}05_{0.00}$	$\mathbf{1.6e\text{-}06}_{0.00}$	$0.481_{0.30}$	$\mathbf{2.3e\text{-}06}_{0.00}$
	Cosine	$0.102_{0.04}$	$\mathbf{2.3e\text{-}06}_{0.00}$	$0.083_{0.02}$	$\mathbf{1.6e\text{-}06}_{0.00}$	$0.860_{0.56}$	$\mathbf{2.3e\text{-}06}_{0.00}$
	Euclidean	$0.064_{0.02}$	$\mathbf{2.3e\text{-}06}_{0.00}$	$3.8e\text{-}05_{0.00}$	$\mathbf{1.6e\text{-}06}_{0.00}$	$0.783_{0.49}$	$\mathbf{2.3e\text{-}06}_{0.00}$
	L1	$0.028_{0.01}$	$\mathbf{2.3e\text{-}06}_{0.00}$	$1.3e\text{-}05_{0.00}$	$\mathbf{1.6e\text{-}06}_{0.00}$	$\mathbf{0.208}_{0.14}$	$\mathbf{2.3e\text{-}06}_{0.00}$
2	Cosine + L1-Norm	$0.358_{0.07}$	$0.673_{0.04}$	$0.248_{0.10}$	$0.668_{0.07}$	$0.301_{0.20}$	$0.261_{0.29}$
	Cosine + L2-Norm	$0.358_{0.07}$	$0.751_{0.04}$	$\mathbf{0.234}_{0.13}$	$0.706_{0.13}$	$0.816_{0.36}$	$0.602_{0.46}$
	Cosine	$\mathbf{0.328}_{0.06}$	$0.957_{0.10}$	$0.301_{0.10}$	$0.950_{0.04}$	$1.049_{0.25}$	$0.920_{0.33}$
	Euclidean	$0.352_{0.06}$	$1.098_{0.06}$	$0.301_{0.08}$	$1.088_{0.04}$	$0.869_{0.43}$	$0.860_{0.36}$
	L1	$0.356_{0.07}$	$\mathbf{0.673}_{0.04}$	$0.265_{0.11}$	$\mathbf{0.636}_{0.05}$	$\mathbf{0.278}_{0.20}$	$\mathbf{0.244}_{0.27}$
4	Cosine + L1-Norm	$0.557_{0.06}$	$\mathbf{1.029}_{0.06}$	$\mathbf{0.427}_{0.09}$	$1.019_{0.05}$	$0.825_{0.35}$	$\mathbf{0.516}_{0.40}$
	Cosine + L2-Norm	$0.576_{0.06}$	$1.095_{0.04}$	$0.521_{0.07}$	$1.079_{0.07}$	$1.231_{0.12}$	$0.881_{0.25}$
	Cosine	$0.595_{0.01}$	$1.184_{0.01}$	$0.461_{0.09}$	$1.093_{0.10}$	$1.132_{0.34}$	$0.933_{0.43}$
	Euclidean	$\mathbf{0.548}_{0.06}$	$1.427_{0.07}$	$0.496_{0.05}$	$1.273_{0.07}$	$1.264_{0.16}$	$1.040_{0.26}$
	L1	$0.565_{0.05}$	$1.032_{0.05}$	$0.428_{0.09}$	$\mathbf{1.014}_{0.05}$	$\mathbf{0.777}_{0.34}$	$0.539_{0.45}$

Table 4-2: Gradient distance function comparison on London Smartmeter for different batch sizes and model architectures (seeds 10, 43, 28)

4-4-2 Quantile Predictions for Batch Size of 1

In the paper, the quantiles in Figure 7 are predicted based on a batch of 4 samples. In the following experiment, we train the gradient inversion model to predict the quantiles of a single batched sample for different model architectures.

In Figure 4-22, the quantiles are plotted, and we can clearly see whether the inversion model can capture the relationship between gradients and observations or targets. For all models with an FCN head, the model is able to predict the quantiles with relatively high detail for the target. However, the model is unable to capture this for the GRU-2-GRU architecture on the London Smartmeter and Proprietary datasets, indicating that the loss of information or many-to-one mapping is not learnable here. However, it can learn a rough idea of what the targets look like for the Electricity 370 dataset. Furthermore, the observation quantiles for all but the FCN architecture do not encapsulate the sequence completely, indicating that the relationship of gradients to observations is more difficult to learn for these architectures.

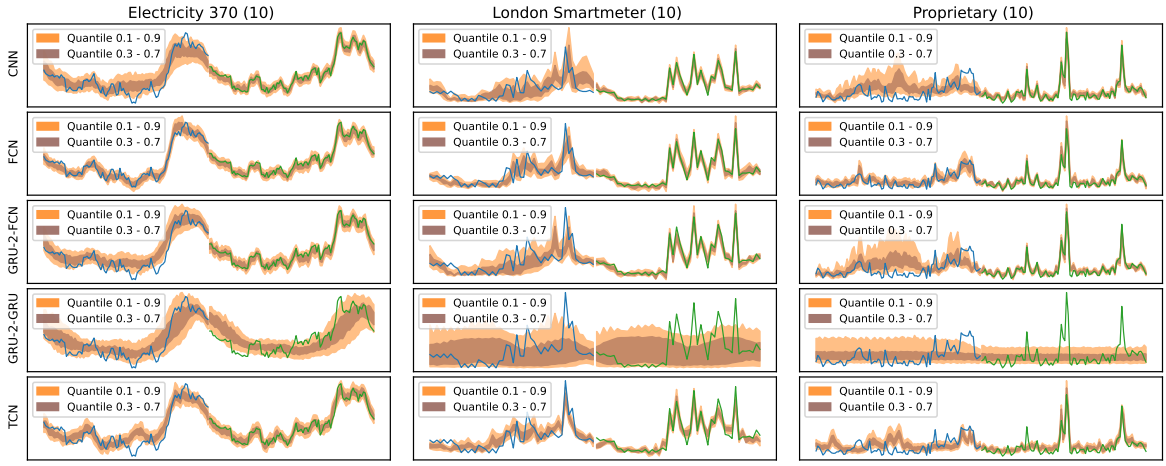


Figure 4-22: Gradient Inversion Model Quantile Predictions for $\mathcal{B} = 1$. Datasets: Electricity 370, London Smartmeter, and Proprietary, (seed 10)

4-4-3 Pinball or Quantile Bounds Regularization

The quantile predictions can be used in at least 2 ways, of which 1 we present in the paper as the Quantile Bounds Regularization ($\mathcal{L}_{\text{bounds}}$). Another method is by using the Pinball loss as regularization ($\mathcal{L}_{\text{pinball}}$). In Table 4-3 the results of both regularization techniques are presented. The columns represent the regularization strength on the observations (λ_{obs}), and the columns present the regularization strength on the targets (λ_{tar}). The Loss represents either using Equation 5 or Equation 11 in the paper, respectively as Pinball loss or Quantile Bounds Loss. The results here for the Quantile Bounds are the same as in the paper and the best are $\lambda_{\text{obs}} = 1$ and $\lambda_{\text{tar}} = 0.1$.

	λ_{obs}	0		0.1		0.5		1	
λ_{tar}	Loss	Observation	Target	Observation	Target	Observation	Target	Observation	Target
0	$\mathcal{L}_{\text{pinball}}$	$\mathbf{0.515}_{0.11}$	$\mathbf{0.068}_{0.05}$	$0.529_{0.11}$	$\mathbf{0.105}_{0.05}$	$0.240_{0.06}$	$\mathbf{0.033}_{0.00}$	$0.207_{0.08}$	$0.063_{0.06}$
	$\mathcal{L}_{\text{bounds}}$	$\mathbf{0.515}_{0.11}$	$\mathbf{0.068}_{0.05}$	$\mathbf{0.504}_{0.12}$	$0.185_{0.11}$	$\mathbf{0.207}_{0.08}$	$0.061_{0.06}$	$\mathbf{0.203}_{0.05}$	$\mathbf{0.060}_{0.03}$
0.1	$\mathcal{L}_{\text{pinball}}$	$\mathbf{0.470}_{0.23}$	$\mathbf{0.017}_{0.01}$	$0.553_{0.09}$	$0.196_{0.11}$	$0.347_{0.08}$	$\mathbf{0.054}_{0.02}$	$0.258_{0.10}$	$0.047_{0.00}$
	$\mathcal{L}_{\text{bounds}}$	$0.567_{0.17}$	$0.048_{0.01}$	$\mathbf{0.449}_{0.14}$	$\mathbf{0.105}_{0.05}$	$\mathbf{0.284}_{0.02}$	$0.237_{0.11}$	$\mathbf{0.144}_{0.03}$	$\mathbf{0.024}_{0.02}$
0.5	$\mathcal{L}_{\text{pinball}}$	$\mathbf{0.468}_{0.18}$	$\mathbf{0.023}_{0.00}$	$0.547_{0.15}$	$0.065_{0.02}$	$0.277_{0.17}$	$\mathbf{0.091}_{0.08}$	$0.224_{0.06}$	$\mathbf{0.047}_{0.01}$
	$\mathcal{L}_{\text{bounds}}$	$0.568_{0.18}$	$0.067_{0.02}$	$\mathbf{0.368}_{0.21}$	$\mathbf{0.026}_{0.02}$	$\mathbf{0.259}_{0.08}$	$0.112_{0.08}$	$\mathbf{0.207}_{0.04}$	$0.131_{0.06}$
1	$\mathcal{L}_{\text{pinball}}$	$0.598_{0.13}$	$0.118_{0.06}$	$\mathbf{0.465}_{0.09}$	$\mathbf{0.070}_{0.05}$	$0.331_{0.13}$	$0.053_{0.02}$	$0.245_{0.05}$	$\mathbf{0.071}_{0.04}$
	$\mathcal{L}_{\text{bounds}}$	$\mathbf{0.514}_{0.07}$	$\mathbf{0.053}_{0.03}$	$0.522_{0.22}$	$0.342_{0.26}$	$\mathbf{0.226}_{0.12}$	$\mathbf{0.040}_{0.03}$	$\mathbf{0.196}_{0.04}$	$0.077_{0.03}$

Table 4-3: TS-Inverse: Comparison of Experiment between Quantile bounds vs Pinball Loss as regularization mechanism. Grid search for the hyperparameter values. Model: TCN, Dataset: Electricity 370, Batch size $\mathcal{B} = 4$, (seeds 10 and 43)

4-4-4 Pure Periodicity Regularization

In the paper, we evaluate the combined effects of the regularization techniques on the TCN architecture. In the following experiments, we present the results of only using the periodicity regularization for the TCN. Furthermore, periodicity as a regularization technique requires fine-tuning the hyperparameter. Table Table 4-4 contains the sMAPE results for the L1-Norm version with different λ_P hyperparameter values, and in Figure 4-23, the reconstructions of the best values are presented. For completeness, Figure 4-24 shows all of the reconstruction results. This figure is quite large but contains the differences between L1 and L2 based reconstructions. The L1 version looked visually better able to regularize the data as it is deemed more robust to outliers, which is possible in this periodicity regularization, as the observations and targets can have different energy load peaks.

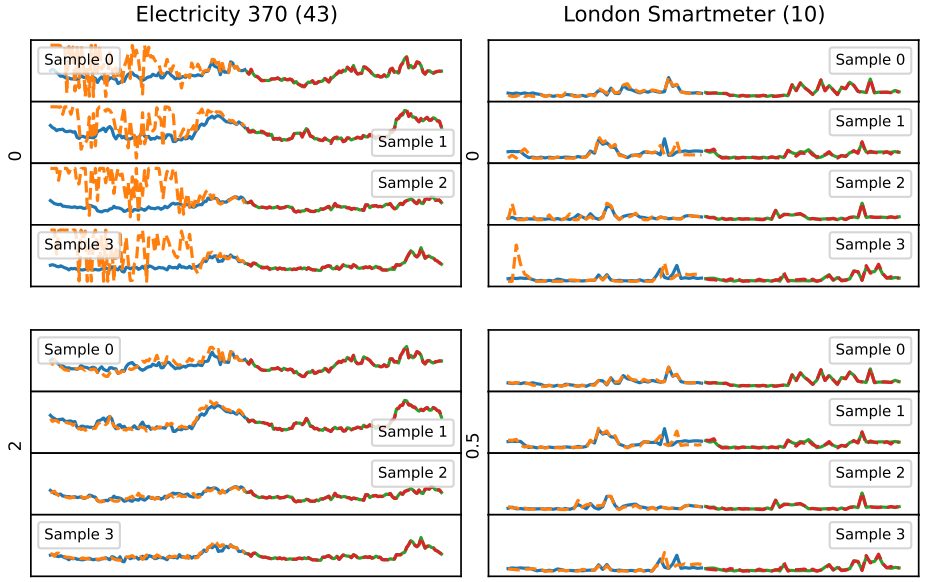


Figure 4-23: TS-Inverse: The best periodicity regularization λ_P compared to no regularization. Rows are λ_P and Columns the dataset. Model: TCN, Datasets: Electricity 370, London Smartmeter, Batch size $\mathcal{B} = 4$, (seeds 10 and 43).

Dataset	Elec. 370		London Smart.		Proprietary	
λ_P	Observation	Target	Observation	Target	Observation	Target
0	$0.515_{0.11}$	$0.068_{0.05}$	$0.603_{0.29}$	$0.250_{0.22}$	$0.956_{0.12}$	$0.197_{0.03}$
0.1	$0.400_{0.04}$	$0.146_{0.12}$	$0.838_{0.19}$	$0.620_{0.22}$	$0.670_{0.02}$	$0.369_{0.07}$
0.5	$0.306_{0.16}$	$0.219_{0.18}$	$\mathbf{0.467}_{0.18}$	$\mathbf{0.250}_{0.21}$	$0.396_{0.02}$	$0.214_{0.03}$
1	$0.355_{0.16}$	$0.375_{0.13}$	$0.714_{0.24}$	$0.538_{0.43}$	$\mathbf{0.372}_{0.01}$	$\mathbf{0.167}_{0.07}$
2	$\mathbf{0.163}_{0.05}$	$\mathbf{0.063}_{0.05}$	$0.658_{0.06}$	$0.289_{0.14}$	$0.452_{0.02}$	$0.193_{0.01}$

Table 4-4: TS-Inverse: Comparing λ_P for periodicity regularization. Model: TCN, Datasets: Electricity 370, London Smartmeter, and Proprietary, Batch size $\mathcal{B} = 4$, (seeds 10 and 43)

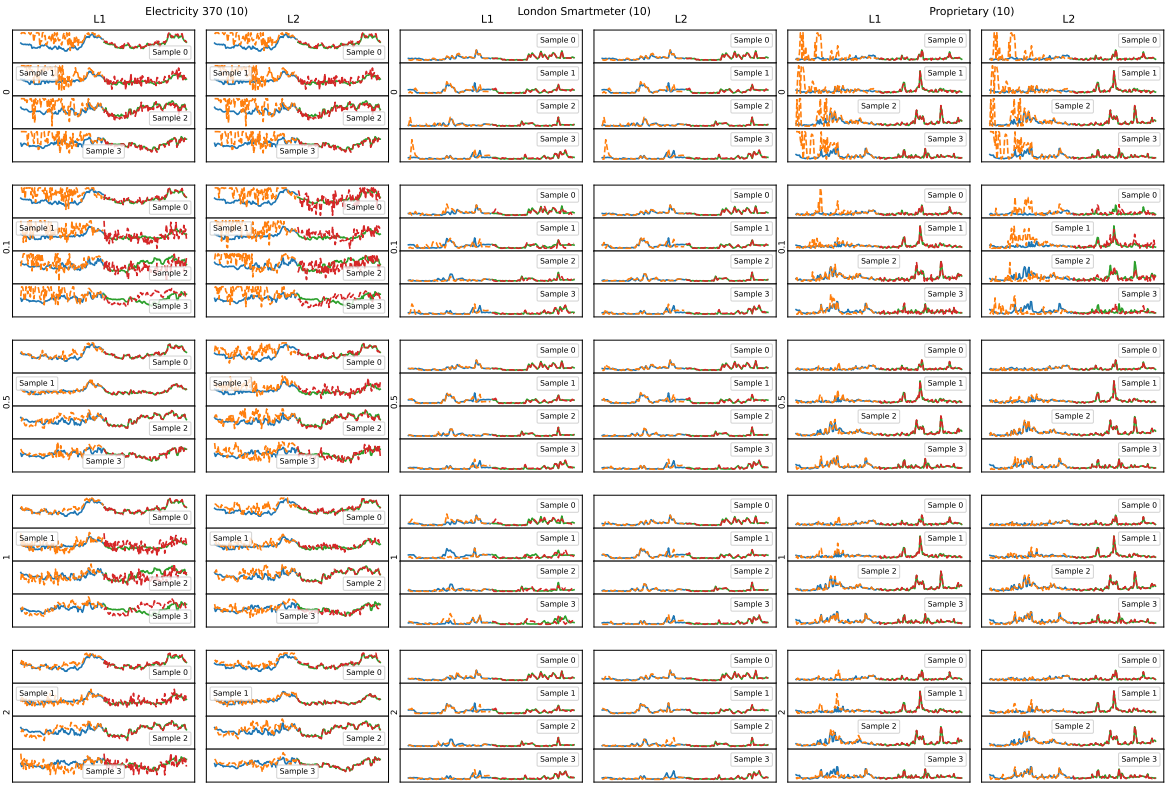


Figure 4-24: TS-Inverse: Reconstructions with Periodicity Regularization. The columns represent different $\lambda_{\text{periodicity}}$ hyperparameter values and the rows represent the loss (L1 or L2). Model: TCN, Datasets: Electricity 370, London Smartmeter, and Proprietary, Batch size $\mathcal{B} = 4$, (seeds 10 and 43)

4-4-5 Pure Trend Regularization

We also evaluated the pure trend regularization technique on the TCN architecture. The trend regularization also requires finetuning the hyperparameter values (λ_T). Table Table 4-5 presents the sMAPE results with different hyperparameter values. The case of λ_T=0 is the same as $\lambda_P = 0$ in Figure 4-23. In Figure 4-25 are the best settings presented for the Electricity 370 and London Smartmeter datasets. For the Electricity 370 dataset we choose $\lambda_T = 2$ because it had the lowest Observation metric, and the targets seemed to be reconstructed reasonably well visually. However, as one can spot in Figure 4-25, the regularization is so strong that the reconstruction converges to a straightly in the Observations. All of the trend regularized reconstructions are presented in Figure 4-26.

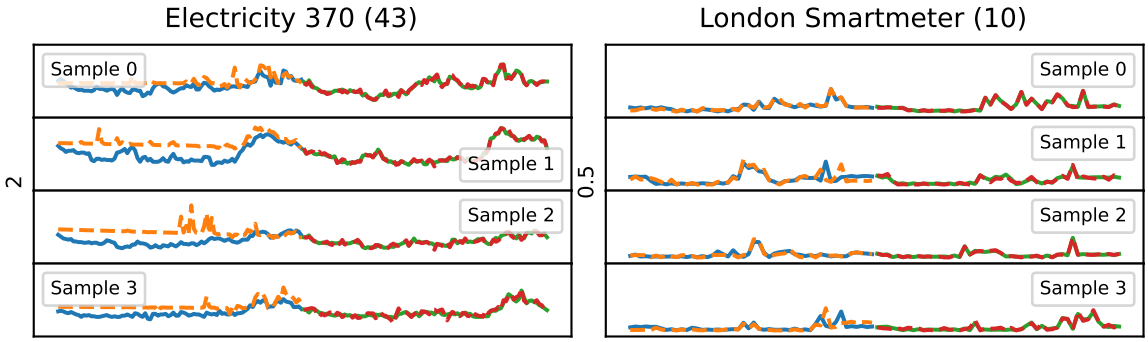


Figure 4-25: TS-Inverse: the best trend regularization λ_T. Model: TCN, Datasets: Electricity 370, London Smartmeter, and Proprietary, Batch size $\mathcal{B} = 4$, (seeds 10 and 43)

Dataset	Elec. 370		London Smart.		Proprietary	
λ_T	Observation	Target	Observation	Target	Observation	Target
0	$0.515_{0.11}$	$0.068_{0.05}$	$0.603_{0.29}$	$0.250_{0.22}$	$0.956_{0.12}$	$0.197_{0.03}$
0.1	$0.505_{0.15}$	$\mathbf{0.054}_{0.02}$	$0.853_{0.08}$	$0.416_{0.06}$	$0.955_{0.02}$	$0.443_{0.27}$
0.5	$0.392_{0.08}$	$0.152_{0.12}$	$\mathbf{0.495}_{0.21}$	$\mathbf{0.247}_{0.23}$	$0.736_{0.10}$	$0.482_{0.30}$
1	$0.419_{0.17}$	$0.324_{0.21}$	$0.685_{0.04}$	$0.568_{0.14}$	$\mathbf{0.475}_{0.12}$	$\mathbf{0.145}_{0.02}$
2	$\mathbf{0.281}_{0.05}$	$0.124_{0.06}$	$0.633_{0.01}$	$0.250_{0.21}$	$0.515_{0.06}$	$0.242_{0.07}$

Table 4-5: TS-Inverse: Comparing λ_T for trend regularization. Model: TCN, Datasets: Electricity 370, London Smartmeter, and Proprietary, Batch size $\mathcal{B} = 4$, (seeds 10 and 43)

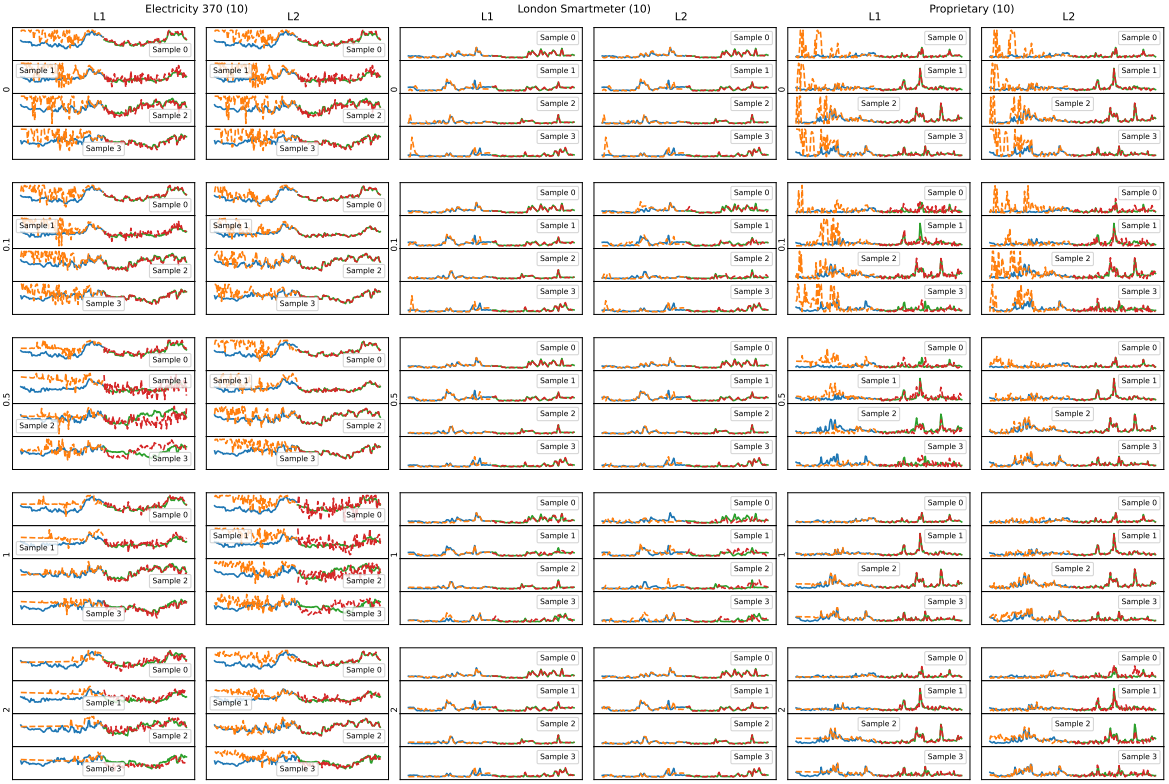


Figure 4-26: TS-Inverse: Reconstructions with Trend Regularization. The columns represent different λ_{trend} hyperparameter values and the rows represent the loss (L1 or L2). Model: TCN, Datasets: Electricity 370, London Smartmeter, and Proprietary, Batch size $\mathcal{B} = 4$, (seeds 10 and 43)

Chapter 5

Conclusion

This thesis investigates the privacy risks associated with federated learning for time series forecasting models, focusing specifically on gradient inversion attacks. The study evaluates the effectiveness of existing gradient on time series forecasting models, proposes tailored improvements for these attacks, and examines the vulnerability of forecasting model architectures on these attacks.

RQ1: How effective are existing gradient inversion attacks for federated time series forecasting situations: The empirical analysis reveals that current gradient inversion attacks, primarily designed for image and text classification, struggle with time series forecasting. The difficulties stem from several factors: forecasting uses regression losses like mean squared error instead of cross-entropy loss. Consequently, existing methods to analytically reconstruct targets fail in time series scenarios. Additionally, time series forecasting attackers aim to reconstruct both observations and targets, unlike in classification tasks where only inputs are private. The analysis has shown that gradient inversion attacks using Cosine Similarity loss as a gradient distance function cannot accurately reconstruct targets. Furthermore, different model architectures in time series forecasting reduce the effectiveness of gradient inversion attacks, as elaborated in **RQ3**. All in all, these key factors make it more difficult for the existing gradient inversion attacks to reconstruct the private time series data.

RQ2: How can gradient inversion attacks be tailored to improve the reconstruction of time series data: To address the challenges in federated time series forecasting from an attacker's perspective, we introduce TS-Inverse, a novel gradient inversion attack for time series forecasting. TS-Inverse features three components: a gradient inversion model mapping gradients to time series quantiles, a loss function combining the L1 norm with periodicity and trend regularization, and the use of quantiles as additional bounds during the inversion optimization.

Additionally, a one-shot analytical technique is proposed for reconstructing targets in regression tasks optimized with the Mean Squared Error loss for a batch size of one. Evaluation results show that TS-Inverse achieves a 2x-10x reduction in reconstruction error compared to existing methods.

RQ3: How does the architecture of a forecasting model influence its vulnerability to gradient inversion attacks: The research examines how different time series forecasting model architectures, such as Gated Recurrent Units (GRU) and Temporal Convolutional Networks (TCN), affect vulnerability to gradient inversion attacks. Findings indicate that recurrent models like GRUs are more privacy-preserving than non-recurrent models due to their design, which selectively forgets and remembers information. Furthermore, the many-to-one mapping from inputs to gradients means each gradient reflects the combined influence of multiple past inputs, complicating the tracing of gradients back to specific inputs, particularly in the GRU-2-GRU architecture. The introduction of dropout mechanisms in TCNs also challenges gradient inversion due to the probabilistic loss of information. However, incorporating this probabilistic nature into the optimization process can effectively mitigate this issue.

Bibliography

[1] N. Ahmad, Y. Ghadi, M. Adnan, and M. Ali, "Load Forecasting Techniques for Power System: Research Challenges and Survey," *IEEE Access*, vol. 10, pp. 71 054–71 090, 2022, 59 citations (Semantic Scholar/DOI) [2024-06-19] 59 citations (Semantic Scholar/DOI) [2024-06-14] 47 citations (Crossref/DOI) [2024-01-17] Conference Name: IEEE Access. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9812604

[2] S. Haben, S. Arora, G. Giasemidis, M. Voss, and D. Vukadinović Greetham, "Review of low voltage load forecasting: Methods, applications, and recommendations," *Applied Energy*, vol. 304, p. 117798, Dec. 2021, 64 citations (Semantic Scholar/DOI) [2024-06-19] 64 citations (Semantic Scholar/DOI) [2024-06-14] 48 citations (Crossref/DOI) [2024-01-17]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261921011326

[3] S. Čaušević, R. Snijders, G. Pingen, P. Pileggi, M. Theelen, M. Warnier, F. Brazier, and K. Kok, "Flexibility prediction in Smart Grids: Making a case for Federated Learning," in *CIRED 2021 - The 26th International Conference and Exhibition on Electricity Distribution*, vol. 2021, Sep. 2021, pp. 1983–1987, 1 citations (Semantic Scholar/DOI) [2024-06-19] 2 citations (Crossref/DOI) [2024-01-17]. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9692324

[4] J. Zhu, H. Dong, W. Zheng, S. Li, Y. Huang, and L. Xi, "Review and prospect of data-driven techniques for load forecasting in integrated energy systems," *Applied Energy*, vol. 321, p. 119269, Sep. 2022, 81 citations (Semantic Scholar/DOI) [2024-06-19]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261922006262

[5] P. Lara-Benítez, M. Carranza-García, J. M. Luna-Romera, and J. C. Riquelme, "Temporal Convolutional Networks Applied to Energy-Related Time Series Forecasting," *Applied Sciences*, vol. 10, no. 7, p. 2322, Jan. 2020, 98 citations (Crossref/DOI) [2024-01-22] Number: 7 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2076-3417/10/7/2322

[6] Y. Wang, D. Gan, M. Sun, N. Zhang, Z. Lu, and C. Kang, "Probabilistic individual load forecasting using pinball loss guided LSTM," *Applied Energy*, vol.

235, pp. 10–20, Feb. 2019, 274 citations (Semantic Scholar/DOI) [2024-06-19] 278 citations (Crossref/DOI) [2024-06-11]. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0306261918316465

[7] S. Čaušević, S. Sharma, S. Ben Aziza, A. van der Veen, and E. Lazovik, "LV grid state estimation using local flexible assets: a federated learning approach," in *27th International Conference on Electricity Distribution (CIRED 2023)*, vol. 2023, Jun. 2023, pp. 1045–1049, 0 citations (Semantic Scholar/DOI) [2024-06-19] 1 citations (Crossref/DOI) [2024-01-17]. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10267781

[8] N. Truong, K. Sun, S. Wang, F. Guitton, and Y. Guo, "Privacy Preservation in Federated Learning: An insightful survey from the GDPR Perspective," Mar. 2021, 9 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:2011.05411 [cs]. [Online]. Available: http://arxiv.org/abs/2011.05411

[9] M. N. Fekri, K. Grolinger, and S. Mir, "Distributed load forecasting using smart meter data: Federated learning with Recurrent Neural Networks," *International Journal of Electrical Power & Energy Systems*, vol. 137, p. 107669, May 2022, 68 citations (Crossref/DOI) [2024-01-17] 49 citations (Crossref) [2023-10-12]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142061521008991

[10] H. Bousbiat, R. Bousselidj, Y. Himeur, A. Amira, F. Bensaali, F. Fadli, W. Mansoor, and W. Elmenreich, "Crossing Roads of Federated Learning and Smart Grids: Overview, Challenges, and Perspectives," Apr. 2023, 5 citations (Semantic Scholar/arXiv) [2024-06-19] 5 citations (Semantic Scholar/DOI) [2024-06-14] arXiv:2304.08602 [cs]. [Online]. Available: http://arxiv.org/abs/2304.08602

[11] X. Cheng, C. Li, and X. Liu, "A Review of Federated Learning in Energy Systems," in *2022 IEEE/IAS Industrial and Commercial Power System Asia (I&CPS Asia)*, Jul. 2022, pp. 2089–2095, 19 citations (Semantic Scholar/DOI) [2024-06-19] 19 citations (Semantic Scholar/DOI) [2024-06-14] 7 citations (Crossref/DOI) [2024-01-17]. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9949863

[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. PMLR, Apr. 2017, pp. 1273–1282, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[13] J. Wen, Z. Zhang, Y. Lan, Z. Cui, J. Cai, and W. Zhang, "A survey on federated learning: challenges and applications," *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, pp. 513–535, Feb. 2023. [Online]. Available: https://doi.org/10.1007/s13042-022-01647-y

[14] L. Lyu, H. Yu, X. Ma, C. Chen, L. Sun, J. Zhao, Q. Yang, and P. S. Yu, "Privacy and Robustness in Federated Learning: Attacks and Defenses," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2022, 221 citations (Semantic Scholar/DOI) [2024-06-19] 220 citations (Semantic Scholar/DOI) [2024-06-14] 36 citations (Crossref/DOI) [2024-01-22] Conference Name: IEEE

Transactions on Neural Networks and Learning Systems. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9945997

[15] N. Rodríguez-Barroso, D. Jiménez-López, M. V. Luzón, F. Herrera, and E. Martínez-Cámara, "Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges," *Information Fusion*, vol. 90, pp. 148–173, Feb. 2023, 104 citations (Semantic Scholar/DOI) [2024-06-19] 101 citations (Semantic Scholar/DOI) [2024-06-14] 71 citations (Crossref/DOI) [2024-06-05]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253522001439

[16] S. Liu, Z. Wang, and Q. Lei, "Data reconstruction attacks and defenses: a systematic evaluation," Feb. 2024, arXiv:2402.09478 [cs]. [Online]. Available: http://arxiv.org/abs/2402.09478

[17] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated Machine Learning: Concept and Applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 12:1–12:19, Jan. 2019, 1867 citations (Semantic Scholar/DOI) [2024-06-19] 1866 citations (Semantic Scholar/DOI) [2024-06-14] 2376 citations (Crossref/DOI) [2024-01-17]. [Online]. Available: https://dl.acm.org/doi/10.1145/3298981

[18] G. Box, "Time series analysis, forecasting and control," in *A Very British Affair: Six Britons and the Development of Time Series Analysis During the 20th Century*, T. C. Mills, Ed. London: Palgrave Macmillan UK, 2013, pp. 161–215. [Online]. Available: https://doi.org/10.1057/9781137291264_6

[19] A. Casolaro, V. Capone, G. Iannuzzo, and F. Camastra, "Deep Learning for Time Series Forecasting: Advances and Open Problems," *Information*, vol. 14, no. 11, p. 598, Nov. 2023, 8 citations (Semantic Scholar/DOI) [2024-06-14] 2 citations (Crossref/DOI) [2024-01-22] Number: 11 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2078-2489/14/11/598

[20] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," Sep. 2020, 882 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:1912.09363 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1912.09363

[21] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986, 24953 citations (Semantic Scholar/DOI) [2024-06-19] 17206 citations (Crossref/DOI) [2024-06-08] Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/323533a0

[22] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, 77369 citations (Semantic Scholar/DOI) [2024-06-19] 55360 citations (Crossref/DOI) [2024-06-08] Conference Name: Neural Computation. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6795963

[23] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," Sep. 2014, 20896 citations (Semantic Scholar/arXiv)

[2024-06-19] 20522 citations (Semantic Scholar/arXiv) [2024-05-01] 68 citations (INSPIRE-HEP/arXiv) [2024-05-01] arXiv:1406.1078 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1406.1078

[24] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," Apr. 2018, 3705 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:1803.01271 [cs]. [Online]. Available: http://arxiv.org/abs/1803.01271

[25] A. Arif, N. Javaid, M. Anwar, A. Naeem, H. Gul, and S. Fareed, "Electricity Load and Price Forecasting Using Machine Learning Algorithms in Smart Grid: A Survey," in *Web, Artificial Intelligence and Network Applications*, ser. Advances in Intelligent Systems and Computing, L. Barolli, F. Amato, F. Moscato, T. Enokido, and M. Takizawa, Eds. Cham: Springer International Publishing, 2020, pp. 471–483, 16 citations (Semantic Scholar/DOI) [2024-06-14] 11 citations (Crossref/DOI) [2024-01-23].

[26] I. Richardson, "Integrated high-resolution modelling of domestic electricity demand and low voltage electricity distribution networks," thesis, Loughborough University, Jan. 2011. [Online]. Available: https://repository.lboro.ac.uk/articles/thesis/Integrated_high-resolution_modelling_of_domestic_electricity_demand_and_low_voltage_electricity_distribution_networks/9520403/1

[27] M. Alhussein, K. Aurangzeb, and S. I. Haider, "Hybrid CNN-LSTM Model for Short-Term Individual Household Load Forecasting," *IEEE Access*, vol. 8, pp. 180 544–180 557, 2020, 186 citations (Crossref/DOI) [2024-01-17] Conference Name: IEEE Access. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9210478

[28] F. U. M. Ullah, A. Ullah, I. U. Haq, S. Rho, and S. W. Baik, "Short-Term Prediction of Residential Power Energy Consumption via CNN and Multi-Layer Bi-Directional LSTM Networks," *IEEE Access*, vol. 8, pp. 123 369–123 380, 2020, 90 citations (Crossref/DOI) [2024-01-17] Conference Name: IEEE Access. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8945363

[29] M. N. Fekri, H. Patel, K. Grolinger, and V. Sharma, "Deep learning for load forecasting with smart meter data: Online Adaptive Recurrent Neural Network," *Applied Energy*, vol. 282, p. 116177, Jan. 2021, 125 citations (Crossref/DOI) [2024-01-17]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261920315804

[30] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, Feb. 1978, 6042 citations (Semantic Scholar/DOI) [2024-06-19] 4188 citations (Crossref/DOI) [2024-06-04] Conference Name: IEEE Transactions on Acoustics, Speech, and Signal Processing. [Online]. Available: https://ieeexplore.ieee.org/document/1163055

[31] D. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," Jul. 1994. [Online]. Available: https://www.semanticscholar.org/paper/Using-Dynamic-Time-Warping-to-Find-Patterns-in-Time-Berndt-Clifford/1ac57524ba2d2a69c1bb6defed7352a06fd7050d

[32] M. Cuturi and M. Blondel, "Soft-DTW: a Differentiable Loss Function for Time-Series," Feb. 2018, 487 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:1703.01541 [stat]. [Online]. Available: http://arxiv.org/abs/1703.01541

[33] Y. Chen, C. Obrecht, and F. Kuznik, "Enhancing peak prediction in residential load forecasting with soft dynamic time wrapping loss functions," *Integrated Computer-Aided Engineering*, vol. 31, no. 3, pp. 327–340, Jan. 2024, 0 citations (Semantic Scholar/DOI) [2024-06-19] 0 citations (Crossref/DOI) [2024-04-29] Publisher: IOS Press. [Online]. Available: https://content.iospress.com/articles/integrated-computer-aided-engineering/ica230731

[34] Y. Wang, N. Gao, and G. Hug, "Personalized Federated Learning for Individual Consumer Load Forecasting," *CSEE Journal of Power and Energy Systems*, vol. 9, no. 1, pp. 326–330, Jan. 2023, 23 citations (Semantic Scholar/DOI) [2024-06-19] 4 citations (Crossref/DOI) [2024-01-17] Conference Name: CSEE Journal of Power and Energy Systems. [Online]. Available: https://ieeexplore.ieee.org/document/9770488

[35] Y. Yang, Z. Wang, S. Zhao, and J. Wu, "An integrated federated learning algorithm for short-term load forecasting," *Electric Power Systems Research*, vol. 214, p. 108830, Jan. 2023, 18 citations (Semantic Scholar/DOI) [2024-06-19] 9 citations (Crossref/DOI) [2024-01-17] 7 citations (Crossref) [2023-10-12]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378779622008835

[36] J. D. Fernández, S. P. Menci, C. M. Lee, A. Rieger, and G. Fridgen, "Privacy-preserving federated learning for residential short-term load forecasting," *Applied Energy*, vol. 326, p. 119915, Nov. 2022, 27 citations (Semantic Scholar/DOI) [2024-06-19] 30 citations (Crossref/DOI) [2024-06-04]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261922011722

[37] V. Perifanis, N. Pavlidis, R.-A. Koutsiamanis, and P. S. Efraimidis, "Federated learning for 5G base station traffic forecasting," *Computer Networks*, vol. 235, p. 109950, Nov. 2023, 5 citations (Crossref/DOI) [2024-03-12]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S138912862300395X

[38] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the Convergence of FedAvg on Non-IID Data," Jun. 2020, 1772 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:1907.02189 [cs, math, stat]. [Online]. Available: http://arxiv.org/abs/1907.02189

[39] Y. Wang, M. Jia, N. Gao, L. Von Krannichfeldt, M. Sun, and G. Hug, "Federated Clustering for Electricity Consumption Pattern Extraction," *IEEE Transactions on Smart Grid*, vol. 13, no. 3, pp. 2425–2439, May 2022, 36 citations (Semantic Scholar/DOI) [2024-06-19] 21 citations (Crossref/DOI) [2024-01-17] 16 citations (Crossref) [2023-10-12] Conference Name: IEEE Transactions on Smart Grid. [Online]. Available: https://ieeexplore.ieee.org/document/9693930

[40] X. Ma, J. Zhang, S. Guo, and W. Xu, "Layer-wised Model Aggregation for Personalized Federated Learning," May 2022, 57 citations (Semantic Scholar/arXiv) [2024-01-17] arXiv:2205.03993 [cs]. [Online]. Available: http://arxiv.org/abs/2205.03993

[41] J. S. Nightingale, Y. Wang, F. Zobiri, and M. A. Mustafa, "Effect of Clustering in Federated Learning on Non-IID Electricity Consumption Prediction," in *2022 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, Oct. 2022, pp. 1–5, 5 citations (Semantic Scholar/DOI) [2024-06-19] 3 citations (Crossref/DOI) [2024-01-17] 1 citations (Crossref) [2023-10-12]. [Online]. Available: https://ieeexplore.ieee.org/document/9960569

[42] Y. L. Tun, K. Thar, C. M. Thwal, and C. S. Hong, "Federated Learning based Energy Demand Prediction with Clustered Aggregation," in *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Jan. 2021, pp. 164–167, 38 citations (Semantic Scholar/DOI) [2024-06-19] 25 citations (Crossref/DOI) [2024-01-17] 20 citations (Crossref) [2023-10-12] ISSN: 2375-9356. [Online]. Available: https://ieeexplore.ieee.org/document/9373194

[43] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three Approaches for Personalization with Applications to Federated Learning," Jul. 2020, 463 citations (Semantic Scholar/arXiv) [2024-06-19] 408 citations (Semantic Scholar/arXiv) [2024-01-17] arXiv:2002.10619 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2002.10619

[44] J. Zhang, Y. Hua, H. Wang, T. Song, Z. Xue, R. Ma, and H. Guan, "FedALA: Adaptive Local Aggregation for Personalized Federated Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 9, pp. 11 237–11 244, Jun. 2023, 11 citations (Crossref/DOI) [2024-01-18] Number: 9. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/26330

[45] H. H. Kumar, K. V R, and M. K. Nair, "Federated K-Means Clustering: A Novel Edge AI Based Approach for Privacy Preservation," in *2020 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, Nov. 2020, pp. 52–56, 25 citations (Semantic Scholar/DOI) [2024-06-19] 10 citations (Crossref/DOI) [2024-01-17]. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9499980?casa_token=8o4X6jplObIAAAAA:BWUoNLnCX5fQ08a58n1ypwpdzk6EoarTlICPnyspAaCkg3ju-wfRgJgn2_li5r3B-ncDT4cP8Q

[46] N. Gholizadeh and P. Musilek, "Federated learning with hyperparameter-based clustering for electrical load forecasting," *Internet of Things*, vol. 17, p. 100470, Mar. 2022, 49 citations (Semantic Scholar/DOI) [2024-06-19] 34 citations (Crossref/DOI) [2024-01-17] 28 citations (Crossref) [2023-10-12]. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2542660521001104

[47] European Parliament and Council of the European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance)," Apr. 2016, legislative Body: EP, CONSIL. [Online]. Available: http://data.europa.eu/eli/reg/2016/679/oj/eng

[48] S. Dayal, D. Alhadidi, A. Abbasi Tadi, and N. Mohammed, "Comparative analysis of membership inference attacks in federated learning," in *Proceedings of the 27th*

International Database Engineered Applications Symposium, ser. IDEAS '23. New York, NY, USA: Association for Computing Machinery, May 2023, pp. 185–192. [Online]. Available: https://dl.acm.org/doi/10.1145/3589462.3589502

[49] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning," in *2019 IEEE Symposium on Security and Privacy (SP)*, May 2019, pp. 739–753, 1139 citations (Semantic Scholar/DOI) [2024-06-19] 672 citations (Crossref/DOI) [2024-06-05] ISSN: 2375-1207. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8835245

[50] F. Mo, A. Borovykh, M. Malekzadeh, H. Haddadi, and S. Demetriou, "Layer-wise Characterization of Latent Information Leakage in Federated Learning," May 2021, 22 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:2010.08762 [cs]. [Online]. Available: http://arxiv.org/abs/2010.08762

[51] W. Zhang, S. Tople, and O. Ohrimenko, "Leakage of Dataset Properties in Multi-Party Machine Learning," Jun. 2021, 56 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:2006.07267 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2006.07267

[52] Z. Wang, Y. Huang, M. Song, L. Wu, F. Xue, and K. Ren, "Poisoning-assisted property inference attack against federated learning," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 4, pp. 3328–3340, Jul. 2023, conference Name: IEEE Transactions on Dependable and Secure Computing. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9851511

[53] L. Zhu, Z. Liu, and S. Han, "Deep Leakage from Gradients," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/60a6c4002cc7b29142def8871531281a-Abstract.html

[54] B. Zhao, K. R. Mopuri, and H. Bilen, "iDLG: Improved Deep Leakage from Gradients," Jan. 2020, 477 citations (Semantic Scholar/arXiv) [2024-06-19] 377 citations (Semantic Scholar/arXiv) [2024-01-22] arXiv:2001.02610 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2001.02610

[55] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting Gradients – How easy is it to break privacy in federated learning?" Sep. 2020, 871 citations (Semantic Scholar/arXiv) [2024-06-19] 695 citations (Semantic Scholar/arXiv) [2024-01-22] arXiv:2003.14053 [cs]. [Online]. Available: http://arxiv.org/abs/2003.14053

[56] Y. Wang, J. Deng, D. Guo, C. Wang, X. Meng, H. Liu, C. Ding, and S. Rajasekaran, "SAPAG: A Self-Adaptive Privacy Attack From Gradients," Sep. 2020, 27 citations (Semantic Scholar/arXiv) [2024-06-19] 20 citations (Semantic Scholar/arXiv) [2024-01-22] arXiv:2009.06228 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2009.06228

[57] J. Zhu and M. Blaschko, "R-GAP: recursive gradient attack on privacy," Mar. 2021, arXiv:2010.07733 [cs]. [Online]. Available: http://arxiv.org/abs/2010.07733

[58] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: image batch recovery via GradInversion," Apr. 2021, arXiv:2104.07586 [cs]. [Online]. Available: http://arxiv.org/abs/2104.07586

[59] J. Jeon, J. Kim, K. Lee, S. Oh, and J. Ok, "Gradient Inversion with Generative Image Prior," Oct. 2021, 101 citations (Semantic Scholar/arXiv) [2024-06-19] 72 citations (Semantic Scholar/arXiv) [2024-01-22] arXiv:2110.14962 [cs]. [Online]. Available: http://arxiv.org/abs/2110.14962

[60] J. Xu, C. Hong, J. Huang, L. Y. Chen, and J. Decouchant, "AGIC: Approximate Gradient Inversion Attack on Federated Learning," in *2022 41st International Symposium on Reliable Distributed Systems (SRDS)*, Sep. 2022, pp. 12–22, 10 citations (Semantic Scholar/DOI) [2024-06-19] 3 citations (Semantic Scholar/DOI) [2024-01-22] 0 citations (Crossref/DOI) [2024-01-17] ISSN: 2575-8462. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9996844

[61] D. Scheliga, P. Mäder, and M. Seeland, "Dropout is NOT All You Need to Prevent Gradient Leakage," Nov. 2022, arXiv:2208.06163 [cs]. [Online]. Available: http://arxiv.org/abs/2208.06163

[62] J. Geng, Y. Mou, Q. Li, F. Li, O. Beyan, S. Decker, and C. Rong, "Improved Gradient Inversion Attacks and Defenses in Federated Learning," *IEEE Transactions on Big Data*, pp. 1–13, 2023, 12 citations (Semantic Scholar/DOI) [2024-06-19] 0 citations (Crossref/DOI) [2024-01-17] Conference Name: IEEE Transactions on Big Data. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10024757?casa_token=DCSUrv98_EIAAAAA:D8ezYpvVTAeOTej-eXRBKkuyQ62_R5Z2WKF4_jnr_OzR8_HnT4sQsolOCcF4rtnP6wtV4FGewg

[63] R. Wu, X. Chen, C. Guo, and K. Q. Weinberger, "Learning To Invert: Simple Adaptive Attacks for Gradient Inversion in Federated Learning," in *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*. PMLR, Jul. 2023, pp. 2293–2303, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v216/wu23a.html

[64] J. Deng, Y. Wang, J. Li, C. Wang, C. Shang, H. Liu, S. Rajasekaran, and C. Ding, "TAG: gradient attack on transformer-based language models," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 3600–3610. [Online]. Available: https://aclanthology.org/2021.findings-emnlp.305

[65] M. Vero, M. Balunovic, D. I. Dimitrov, and M. Vechev, "TabLeak: Tabular Data Leakage in Federated Learning," in *Proceedings of the 40th International Conference on Machine Learning*. PMLR, Jul. 2023, pp. 35 051–35 083, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v202/vero23a.html

[66] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, Nov. 1992, 15214 citations (Semantic Scholar/DOI) [2024-06-19] 10721 citations (Crossref/DOI) [2024-03-05]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/016727899290242F

[67] Y. Zheng, "Dropout against Deep Leakage from Gradients," Nov. 2022, 2 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:2108.11106 [cs]. [Online]. Available: http://arxiv.org/abs/2108.11106

[68] J. Geng, Y. Mou, F. Li, Q. Li, O. Beyan, S. Decker, and C. Rong, "Towards General Deep Leakage in Federated Learning," Jan. 2022, 42 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:2110.09074 [cs]. [Online]. Available: http://arxiv.org/abs/2110.09074

[69] Y. Zhang, J. Hare, and A. Prügel-Bennett, "Deep Set Prediction Networks," Apr. 2020, 100 citations (Semantic Scholar/arXiv) [2024-06-19] arXiv:1906.06565 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1906.06565

[70] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," Dec. 2014, arXiv:1409.3215 [cs]. [Online]. Available: http://arxiv.org/abs/1409.3215

[71] S. Makridakis, "Accuracy measures: theoretical and practical concerns," *International Journal of Forecasting*, vol. 9, no. 4, pp. 527–529, Dec. 1993. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0169207093900793