ON UNLEARNING FOR TIME-SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-series forecasting is widely used in healthy, transportation, finance, and many other domains. Models in these settings are often trained on longitudinal user or entity level records, which may later require removal because they contain sensitive or proprietary information or have been corrupted by sensor failures. To address such deletion requests without costly retraining, machine unlearning has been widely studied as a practical mechanism for privacy protection and data governance. However, the application of machine unlearning to time series prediction has not yet been well realized, it mainly dues to the following unique challenges: Gradient-based unlearning can be unstable because a deleted observation participates in multiple causally connected forecasting windows, causing parameter updates to propagate beyond the requested interval and degrade retained forecasting utility. Label-guided updating offers a more controlled alternative, but continuous and context-dependent forecasts lack a suitable replacement target, while the exact-retrained output is unavailable during unlearning. Moreover, the remaining support for a deleted temporal pattern is highly non-uniform. Some affected windows retain structurally similar counterparts in the retained data, whereas others become underrepresented or isolated. We present **RDTU**, a **R**esidual **D**iffusion framework of **T**ime-series **U**nlearning that constructs labels without accessing an exact-retrained model at unlearning time. RDTU first uses a retained-set neural tangent kernel predictor to obtain a deletion-compatible base forecast. Then it quantifies the global and local structural support of each affected window using the volume contribution of the retained-reference data. Conditioned on the query window, retained-set prediction, and structural support, a diffusion model generates a residual correction that estimates the counterfactual forecast, yielding a pseudo-label field that guides a lightweight model update. Experiments across real-world forecasting benchmarks show that RDTU consistently produces unlearned models that most closely match exact retraining among the evaluated efficient unlearning methods. It achieves this alignment while maintaining retained forecasting utility and reducing membership-inference signals.

1 INTRODUCTION

Time-series forecasting has become a fundamental component of data-driven decision making across finance, healthcare, transportation, energy systems, and intelligent manufacturing, where historical observations are used to anticipate future risks, demands, and system states (Lim & Zohren, 2021; Zonta et al., 2020). As forecasting models are trained on increasingly large and continuously collected data streams, they also face growing demands for selective data removal. In privacy-sensitive domains such as finance and healthcare, training sequences may contain sensitive personal information; under Article 17 of the European Union's General Data Protection Regulation (GDPR), data subjects may request the erasure of their personal data under specified conditions (EU, 2016). A parallel need arises in industrial monitoring, where sensor failures, calibration drift, and communication faults can introduce erroneous observations, while equipment aging can render previously collected operating patterns obsolete (Teh et al., 2020; Goknil et al., 2023). Once incorporated into model training, such observations may distort the learned temporal dependencies and compromise subsequent forecasts.

To meet these deletion requirements, machine unlearning aims to remove the influence of designated training data from a trained model (Cao & Yang, 2015). Existing approaches are broadly divided

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

into exact and approximate unlearning. Exact unlearning seeks to match the distribution of models retrained from scratch on the retained data, but often incurs prohibitive computational costs (Ginart et al., 2019). Approximate unlearning instead updates the original model to efficiently emulate this behavior. Gradient-ascent methods increase the forgetting-set loss, but their lack of an explicit convergence target can cause unstable optimization and severe utility degradation. NegGrad+ (Kurmanji et al., 2023) and NPO (Zhang et al., 2024) mitigate this problem by balancing gradients or bounding the forgetting objective. However, they primarily control how the model departs from forgotten samples rather than specifying its desired post-deletion behavior. This motivates label-based methods such as random labeling, which replace forgotten targets with constructed alternatives (Graves et al., 2021; Di et al., 2026).

Methods developed for image and text unlearning do not transfer directly to time-series forecasting. Forecasting windows are ordered and extensively overlapping, so gradient ascent on forgotten windows may disrupt temporal representations shared with retained data. Label-based methods also lack a natural replacement target because arbitrary continuous values may violate temporal and cross-variable structures. Unlike classification labels or abstention responses (Graves et al., 2021; Maini et al., 2024), time series have no universal rejection label. TS-Unlearn addresses this problem by feeding Gaussian inputs into the original model to construct forgetting targets and dynamically balancing forgetting and retention (Wang et al., 2026). However, these targets are not conditioned on the specific query or its local support in the retained data. They constrain outputs to a plausible range but cannot approximate the predictions of exact unlearning.

To address these challenges, we adopt label-based approximate unlearning and construct labels that approximate the predictions of exact unlearning for time-series forecasting. Because obtaining such labels directly would require retraining on the strictly retained set, we use a lightweight neural tangent kernel (NTK) predictor as an efficient regression surrogate. However, finite-width networks perform feature learning and nonlinear optimization, creating a non-negligible gap between their predictions and the NTK approximation. We therefore introduce training a small conditional diffusion model to learn the residual distribution between the trained model and its NTK predictor and transfer this residual pattern to retained-set conditions. To avoid conditioning on high-dimensional raw time-series windows, we further propose the structural irrelevance score (SIS), which summarizes complementary global directional change and local retained support. Conditioned on the retained-set NTK prediction and SIS, diffusion generates a residual correction that forms a suitable label for each affected window. Extensive experiments on univariate and multivariate datasets demonstrate a favorable balance among unlearning effectiveness, forecasting utility, and computational efficiency. Our contributions can be summarized as follows:

- We formulate time-series forecasting unlearning under strict temporal deletion closure, which identifies all forecasting windows affected by a deletion request, and cast efficient approximate unlearning as the construction of counterfactual labels that approximate the predictions of an exact-retrained model.
- We introduce the Structural Irreplaceability Score (SIS), which complements the model-dependent NTK prediction by quantifying the global and local structural support of each affected window. SIS provides a compact condition for residual generation and reduces reliance on costly raw-window representations.
- We propose **RDTU**, a label-guided Residual Diffusion framework that combines a retained-set NTK predictor with a simple conditional residual diffusion model. The NTK predictor provides a lightweight counterfactual base prediction, while the diffusion model compensates for the substantial finite-width discrepancy between the NTK approximation and the corresponding trained neural predictor.

2 Problem Definition and Related Works

2.1 Sliding-Window Forecasting

Let $\{x_1, \ldots, x_i, \ldots, x_T\}$ denote the observed time series to train a forecasting model [1]. Given a lookback length ℓ and a forecasting horizon h, each valid forecast origin t induces a supervised

[1] We focus on univariate data to simplify notation, and the concepts can be extended to multivariate ones.

forecasting sample $\boldsymbol{x}_t = (x_{t-\ell}, ..., x_{t-1})$ and $\boldsymbol{y}_t = (x_t, ..., x_{t+h-1})$. The original training set is $\mathcal{D} = \{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}_{t \in \mathbb{T}_{\text{train}}}$, where $\mathbb{T}_{\text{train}} \subseteq \{\ell + 1, \ldots, T - h + 1\}$ is the set of valid training forecast origins. An original forecasting model is trained as

$$\boldsymbol{\theta}^*_{\mathcal{D}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \mathcal{D}), \qquad \widehat{\boldsymbol{y}}_t = f_{\boldsymbol{\theta}^*_{\mathcal{D}}}(\boldsymbol{x}_t). \tag{1}$$

where $\mathcal{L}(\boldsymbol{\theta}, \mathcal{D})$ denotes the empirical forecasting loss, which we usually instantiate as the mean squared error between the predicted and ground-truth future windows (Wu et al., 2021).

2.2 Time-series Exact Unlearning and Approximate Unlearning

Consider an unlearning request for a contiguous temporal interval $\mathbb{T}_{\text{f}} = [t_{\text{i}}, t_{\text{s}})$ in the original time series. The request asks the model to forget the training influence of the observations $x_{t_{\text{i}}}, x_{t_{\text{i}}+1}, \ldots, x_{t_{\text{s}}-1}$. For time-series forecasting, this unlearning request cannot be handled by simply removing the training windows whose prediction targets lie inside $\mathbb{T}_{\text{f}}$. Each supervised window contains both an input context and a future prediction target. Therefore, an observation in $\mathbb{T}_{\text{f}}$ may appear as the target of one training window, while also appearing as part of the input context of several subsequent windows.

For a forecasting sample $(\boldsymbol{x}_t, \boldsymbol{y}_t)$, its complete temporal footprint is $\mathbb{I}_t = [t - \ell, t + h)$, which includes its input interval $[t - \ell, t)$ and target interval $[t, t + h)$. We regard a sample as affected by the deletion request whenever any part of its temporal footprint overlaps the requested interval:

$$\mathcal{D}_{\text{f}}(\mathbb{T}_{\text{f}}) = \{(\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathcal{D} \mid \mathbb{I}_t \cap \mathbb{T}_{\text{f}} \neq \varnothing\}. \tag{2}$$

We call $\mathcal{D}_{\text{f}}(\mathbb{T}_{\text{f}})$ the *strict temporal deletion closure* of $\mathbb{T}_{\text{f}}$, *i.e.*, unlearning set. Accordingly, the retained training set is not obtained by deleting only the raw observations in $\mathbb{T}_{\text{f}}$. Instead, it removes every supervised forecasting window affected by those observations: $\mathcal{D}_{\text{r}}(\mathbb{T}_{\text{f}}) = \mathcal{D} \setminus \mathcal{D}_{\text{f}}(\mathbb{T}_{\text{f}})$.

Exact Unlearning. After identifying all windows affected by the deletion request through strict temporal deletion closure, exact unlearning aims to produce a model whose distribution is indistinguishable from that of a model trained without ever observing the forgotten windows (Guo et al., 2020). The canonical realization of this objective is exact retraining. Specifically, it needs to instantiate a new forecasting model with the same architecture as the original model, randomly initialize its parameters from the original initialization distribution, and train it from scratch using only the strictly retained set (Ginart et al., 2019).

Let $\boldsymbol{\theta}_{\mathcal{D}_{\text{r}},0}$ denote the newly initialized parameters, and let $\mathbf{z}_{\mathcal{D}_{\text{r}}}$ collect the remaining training randomness, such as the mini-batch order and stochastic regularization. The exact-retrained model and its prediction are defined as

$$\boldsymbol{\theta}_{\mathcal{D}_{\text{r}},0} \sim p_{\text{init}}, \qquad \mathbf{z}_{\mathcal{D}_{\text{r}}} \sim p_{\text{rng}}, \qquad \boldsymbol{\theta}^*_{\mathcal{D}_{\text{r}}} = \mathcal{A}(\mathcal{D}_{\text{r}}; \boldsymbol{\theta}_{\mathcal{D}_{\text{r}},0}, \mathbf{z}_{\mathcal{D}_{\text{r}}}). \tag{3}$$

where $\mathcal{A}$ denotes the original training algorithm, p_{init} denotes the parameter-initialization distribution, and p_{rng} denotes the joint distribution of the remaining training randomness, including mini-batch sampling and stochastic regularization.

Approximate Unlearning. Conducting a complete model retraining or training an exact unlearning model might be inefficient and costly in many cases. Given the original model $f_{\boldsymbol{\theta}^*_{\mathcal{D}}}$ and a deletion request $\mathbb{T}_{\text{f}}$, an unlearning algorithm produces $\boldsymbol{\theta}_{\text{u}} = \mathcal{U}(\boldsymbol{\theta}^*, \mathcal{D}, \mathbb{T}_{\text{f}})$ and aims to make its forecasting behavior close to that of Exact Retrain:

$$\mathbb{E}_{\boldsymbol{x} \sim P_{\mathbb{Q}}}\left[\left\| f_{\boldsymbol{\theta}_{\text{u}}}(\boldsymbol{x}) - f_{\boldsymbol{\theta}^*_{\text{r}}}(\boldsymbol{x}) \right\|_{\text{F}}^2\right] \approx 0. \tag{4}$$

where $P_{\mathbb{Q}}$ denotes a distribution of permissible forecasting queries.

Gradient-Ascent-Based Approximate Unlearning. Gradient-ascent-based methods increase the training loss on the forgetting set $\mathcal{D}_{\text{f}}$, thereby weakening the associations learned from the samples to be removed (Kurmanji et al., 2023). To preserve the remaining knowledge, gradient ascent on the forgetting set is commonly combined with gradient descent on the retained set $\mathcal{D}_{\text{r}}$. The resulting optimization objective can be written as

$$\boldsymbol{\theta}^{(0)}_{\text{u,GA}} = \boldsymbol{\theta}^*_{\mathcal{D}}, \qquad \boldsymbol{\theta}_{\text{u,GA}} = \arg\min_{\boldsymbol{\theta}} \left[\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}_{\text{r}}) - \lambda_{\text{f}} \mathcal{L}(\boldsymbol{\theta}, \mathcal{D}_{\text{f}})\right]. \tag{5}$$

where λ_{f} controls the strength of forgetting. The negative forgetting-set term reverses the ordinary learning direction and therefore increases the prediction loss on the affected samples. Although this strategy is computationally efficient, directly maximizing the forgetting loss can make the optimization unstable. In particular, aggressive or prolonged gradient ascent may push the model parameters far away from the original trained solution, causing catastrophic degradation on the retained set or even model collapse (Zhang et al., 2024).

Random-Labeling-Based Approximate Unlearning. Random-labeling-based methods replace the original targets of the forgetting samples with randomly generated incorrect targets (Graves et al., 2021). Let $\boldsymbol{y}_t^{\mathrm{rand}}$ denote a random target assigned to an affected window $\boldsymbol{x}_t$. The original model is then fine-tuned using the retained samples and the randomly relabeled forgetting samples:

$$\boldsymbol{\theta}_{\mathrm{u,RL}}^{(0)} = \boldsymbol{\theta}_{\mathcal{D}}^{*}, \qquad \boldsymbol{\theta}_{\mathrm{u,RL}} = \arg\min_{\boldsymbol{\theta}} \left[\mathcal{L}\left(\boldsymbol{\theta}, \mathcal{D}_{\mathrm{r}}\right) + \frac{\lambda_{\mathrm{f}}}{|\mathcal{D}_{\mathrm{f}}|} \sum_{(\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathcal{D}_{\mathrm{f}}} \left\| f_{\boldsymbol{\theta}}\left(\boldsymbol{x}_t\right) - \boldsymbol{y}_t^{\mathrm{rand}} \right\|_{\mathrm{F}}^2 \right]. \tag{6}$$

By associating the affected inputs with unrelated targets, random labeling weakens the original input–target relationship without retraining the model from scratch.

2.3 Neural Tangent Kernel

The neural tangent kernel (NTK) describes the local training dynamics of a neural network through the gradients of its outputs with respect to its parameters (Jacot et al., 2018). Let $f_{\boldsymbol{\theta}} : \mathbb{R}^l \to \mathbb{R}^h$ denote a forecasting model with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$, where l is the input-window length, h is the forecasting horizon. At an initialization $\boldsymbol{\theta}_0$, the output Jacobian is defined as

$$\boldsymbol{J}_{\boldsymbol{\theta}_0}(\boldsymbol{x}) = \left. \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}^{\top}} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} \in \mathbb{R}^{h \times d}. \tag{7}$$

The empirical NTK between two input windows $\boldsymbol{x}$ and $\boldsymbol{x}'$ is the output-space kernel

$$\boldsymbol{K}_{\boldsymbol{\theta}_0}(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{J}_{\boldsymbol{\theta}_0}(\boldsymbol{x}) \boldsymbol{J}_{\boldsymbol{\theta}_0}(\boldsymbol{x}')^{\top} \in \mathbb{R}^{h \times h}. \tag{8}$$

Given a supervised dataset $\mathbb{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$, let $\boldsymbol{K}_{\mathbb{S}\mathbb{S}} \in \mathbb{R}^{h \times h}$ be the block Gram matrix formed by $\boldsymbol{K}_{\boldsymbol{\theta}_0}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, let $\mathbf{k}_{\mathbb{S}}(\boldsymbol{x}) \in \mathbb{R}^{h \times h}$ denote the corresponding cross-kernel matrix, and let $\boldsymbol{y}_{\mathbb{S}} \in \mathbb{R}^h$ contain the stacked vectorized targets. The regularized NTK predictor is defined as

$$g_{\mathbb{S}}(\boldsymbol{x}) = \mathbf{k}_{\mathbb{S}}(\boldsymbol{x}) \left(\boldsymbol{K}_{\mathbb{S}\mathbb{S}} + \lambda_{\mathrm{ntk}} \boldsymbol{I}\right)^{-1} \boldsymbol{y}_{\mathbb{S}}, \tag{9}$$

where $\lambda_{\mathrm{ntk}} > 0$ is a ridge regularization coefficient.

In the infinite-width and linearized-training regime, the NTK predictor characterizes the behavior of a gradient-trained neural network (Lee et al., 2019). In practical finite-width networks, however, nonlinear feature learning, stochastic optimization, and finite-kernel approximations generally cause the NTK prediction to differ from the prediction of the corresponding trained neural model (Lee et al., 2020). For a model trained on $\mathbb{S}$, this discrepancy can be expressed as

$$\boldsymbol{R}_t^{\mathbb{S}} = f_{\boldsymbol{\theta}_{\mathbb{S}}^{*}}\left(\boldsymbol{x}_t\right) - g_{\mathbb{S}}\left(\boldsymbol{x}_t\right). \tag{10}$$

3 Method

Proposed Approach Overview. To achieve the approximate unlearning of time series, we propose RDTU, a **R**esidual **D**iffusion framework of **T**ime-series **U**nlearning. As illustrated in Figure 1, the overall method consists of four main components. First, a retained-set neural tangent kernel (NTK) predictor provides a deletion-compatible base prediction that enables an approximation of the target exact unlearning model. Second, the Structural Irreplaceability Score (SIS) complements the influence estimation by quantifying the global and local structural support provided by each closure window, thereby characterizing the extent to which the unlearned data are replaceable for prediction. Third, conditioned on the NTK prediction and the SIS-based valuation of the forgetting windows, a diffusion model learns the residual between the NTK prediction and the model prediction, generating a refined label field for unlearning. Finally, the generated label field guides the unlearning update, enabling effective removal of the forgetting windows while preserving forecasting performance on strictly retained windows.

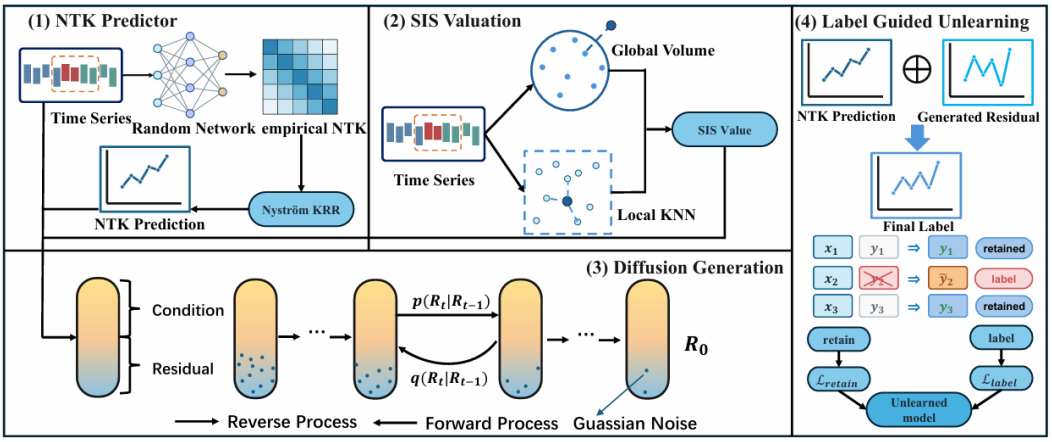


Figure 1: Overview of our proposed time-series unlearning framework RDTU. (1) The retained-set NTK predictor produces a base prediction for the affected window. (2) SIS integrates global directional contribution and local retained support to quantify the structural irreplaceability. (3) Conditioned on the NTK prediction and SIS, the residual diffusion model generates a residual correction. (4) The final label guides the unlearning update.

3.1 NTK Predictor Based Label Construction

Random labeling provides a practical route for approximate unlearning: rather than preserving the original targets of deleted samples, the unlearned model is guided toward constructed targets that reflect the desired post-deletion behaviour. In categorical tasks, such targets are usually formed by replacing the ground-truth label with an alternative label or a prescribed distribution (Chundawat et al., 2023; Di et al., 2026). This option is not directly available for time-series forecasting, where each target is a continuous future trajectory whose value is correlated to its temporal context. Assigning an arbitrary value, such as zero or a retained target, does not in general represent the prediction of a model that has never been trained on the deleted interval.

Follow the definition of approaximate unlearning in Section 2.2, the ideal label for each closure window should be the prediction of the exact-unlearned model. Let $\boldsymbol{\theta}_{\mathrm{r}}^{*}$ denote the parameters obtained by training the forecasting model from scratch on the strict retained set $\mathcal{D}_{\mathrm{r}}$. For each closure query $(\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathcal{D}_{\mathrm{f}}$, the ideal target is $\boldsymbol{y}_t^{*} = f_{\boldsymbol{\theta}_{\mathrm{r}}^{*}}(\boldsymbol{x}_t)$.

Obtaining $\boldsymbol{y}_t^{*}$ from exact unlearned model incurs the substantial computational cost. Therefore, we constructed an NTK predictor as a basic predictor that can be processed, in order to closely approximate our target model with lower cost.

Let $\boldsymbol{\theta}^{(n)}$, $n = 1, \ldots, N$, be N independently sampled initial parameter vectors of the forecasting architecture. For each initialization, we define the empirical neural tangent kernel as

$$K^{(n)}(\boldsymbol{x}, \boldsymbol{x}') = \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}^{(n)}}(\boldsymbol{x})^{\top} \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}^{(n)}}(\boldsymbol{x}'). \tag{11}$$

This kernel measures the similarity between two input windows through the alignment of their parameter gradients at the random initialization. Windows with similar local parameter responses receive large kernel values and can therefore share predictive information through kernel regression.Then we construct a kernel ridge predictor around the random initialization output:

$$\boldsymbol{g}_{\mathbb{S}}^{(n)}(\boldsymbol{x}) = f_{\boldsymbol{\theta}^{(n)}}(\boldsymbol{x}) + \boldsymbol{k}_{\mathbb{S}}^{(n)}(\boldsymbol{x}) \left(\boldsymbol{K}_{\mathbb{S}\mathbb{S}}^{(n)} + \lambda_{\mathrm{ntk}} \boldsymbol{I}\right)^{-1} (\boldsymbol{y}_{\mathbb{S}} - f_{\boldsymbol{\theta}^{(n)}\mathbb{S}}). \tag{12}$$

In Equation 12, $\boldsymbol{K}_{\mathbb{S}\mathbb{S}}^{(n)}$ and $\boldsymbol{k}_{\mathbb{S}}^{(n)}(\boldsymbol{x})$ encode similarities among the available windows and between the query and those windows, respectively. Here, $\boldsymbol{I}$ is an identity matrix matching the kernel matrix. The vectors $\boldsymbol{y}_{\mathbb{S}}$ and $\boldsymbol{f}_{\boldsymbol{\theta}^{(n)}\mathbb{S}}$ stack vectorized targets and predictions $\boldsymbol{f}_{\boldsymbol{\theta}^{(n)}}(\boldsymbol{x}_t)$ in the same sample order. Their difference is the initialization residual. The ridge coefficient stabilizes the regression. For computational efficiency, we approximate these kernel quantities using Nyström features (Williams & Seeger, 2000), avoiding construction of the full empirical NTK matrix while preserving the same

kernel-ridge objective. We further average predictors from multiple independent random initializations to reduce initialization-specific variation (Lakshminarayanan et al., 2017):

$$g_{\mathbb{S}}(\boldsymbol{x}) = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{g}_{\mathbb{S}}^{(n)}(\boldsymbol{x}). \tag{13}$$

Then we can constract NTK predictors of the original training set $\mathcal{D}$ or the strictly retained set $\mathcal{D}_{\mathrm{r}}$ respectively. However, the NTK predictor provides only a linearized approximation of neural-network training. It cannot fully capture finite-width feature learning, nonlinear parameter interactions, or the stochastic optimization dynamics of the actual model (Chizat et al., 2019; Yang & Hu, 2021). Consequently, its predictions exhibit a substantial and non-negligible discrepancy from those of the trained neural predictor. For the original model and its corresponding NTK predictor, we define this prediction gap as the residual:

$$\boldsymbol{R}_t^{\mathcal{D}} = f_{\boldsymbol{\theta}^*}(\boldsymbol{x}_t) - g_{\mathcal{D}}(\boldsymbol{x}_t). \tag{14}$$

A straightforward approach to residual correction is to fit a deterministic regressor between the available conditions and the residuals. However, such methods impose a restricted functional form and generally produce only a point estimate, which may be insufficient for capturing nonlinear or heteroscedastic. We therefore employ a conditional diffusion model to learn the distribution of residual corrections. Importantly, this construction is agnostic to the architecture of the original forecasting model. It does not require access to model-specific hidden representations, gradients, or parameter structures, but relies on the NTK prediction and the features of the corresponding data.

The source residual defined in Equation 14 provides supervision for this process, enabling the diffusion model to capture the finite-width discrepancy between the trained neural predictor and its NTK approximation. Because a deletion request typically removes only a small portion of the training data, we assume that it induces a limited shift in the overall temporal-pattern distribution. Consequently, the residual patterns learned from the complete training set remain transferable to the strictly retained-set conditions, and thus we can obtain the residuals $\widehat{R}_t^{D_r}$ on the retained dataset. For each closure window, the diffusion model receives the window $\boldsymbol{x}_t \in \mathbb{R}^{\ell}$ and the corresponding NTK prediction $g(\boldsymbol{x}_t)$ as conditional information: $\boldsymbol{c}_t = [\boldsymbol{x}_t, g(\boldsymbol{x}_t)]$.

The forward variance schedule and cumulative product are defined as $\alpha_s = 1 - \beta_s$ and $\bar{\alpha}_s = \prod_{q=1}^{s} \alpha_q$ respectively. Here, $\bar{\boldsymbol{r}}_t^{\mathcal{D}} = (\mathrm{vec}(\boldsymbol{R}_t^{\mathcal{D}}) - \mu_r \mathbf{1})/\sigma_r \in \mathbb{R}^{hm}$ is the standardized vectorized source residual. For a randomly sampled diffusion step, Gaussian noise is added to the clean source residual according to the predefined variance schedule:

$$\mathbf{r}_{t,s} = \sqrt{\bar{\alpha}_s}\bar{\boldsymbol{r}}_t^{\mathcal{D}} + \sqrt{1-\bar{\alpha}_s}\epsilon, \qquad \epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{15}$$

The denoiser receives the noisy residual, the diffusion step, and the conditional information, and is trained to predict the injected noise:

$$\mathcal{L}_{\mathrm{diff}} = \mathbb{E}\left[\left\|\epsilon - \boldsymbol{\epsilon}_{\boldsymbol{\psi}}(\mathbf{r}_{t,s}, s, \boldsymbol{c}_t)\right\|_2^2\right]. \tag{16}$$

3.2 Structural Irreplaceability Score Valuation

Training diffusion on raw data is highly expensive, with costs comparable to retraining on the retained set. Using raw data as the conditional input for the diffusion model is to extract the features of the corresponding data and generate the corresponding residual values. Since the NTK prediction values have to some extent already reflected the impact of this data on the entire model, we introduce the Structural Irreplaceability Score (SIS), a compact reference-based descriptor that quantifies how difficult each closure query is to replace using the available reference patterns. SIS combines a global criterion and a local criterion to measure the replaceability of data.

We quantify global replaceability by measuring the change of Volume value in the directional structure of the reference dataset induced by a closure query. Each forecasting window is first converted into a standardized vector representation $\boldsymbol{\phi}_t = \boldsymbol{x}_t$.

We stack the representations of all windows in the current reference dataset $\mathbb{S}$ into the reference matrix $\boldsymbol{\Phi}_{\mathbb{S}}$. Its regularized second-moment matrix summarizes the directions covered by the corresponding temporal-pattern space:

$$\boldsymbol{G}_{\mathbb{S}} = \boldsymbol{\Phi}_{\mathbb{S}}^{\top}\boldsymbol{\Phi}_{\mathbb{S}} + \lambda_{v,\mathbb{S}}\boldsymbol{I}_{d_x}, \qquad \lambda_{v,\mathbb{S}} = \epsilon_v \frac{\mathrm{Tr}\left(\boldsymbol{\Phi}_{\mathbb{S}}^{\top}\boldsymbol{\Phi}_{\mathbb{S}}\right)}{d_x}. \tag{17}$$

The log-determinant of the phase-dependent reference matrix measures the volume covered by the corresponding temporal-pattern space. For a closure query, we define its global contribution as the absolute volume change caused by removing its representation from the complete training set or adding its representation to the strictly retained set:

$$v_t^{\text{global}} = \frac{1}{2}\left|\log\det\left(\boldsymbol{G}_{\mathbb{S}} + \eta_{\mathbb{S}}\boldsymbol{\phi}_t\boldsymbol{\phi}_t^\top\right) - \log\det\left(\boldsymbol{G}_{\mathbb{S}}\right)\right| = \frac{1}{2}\left|\log\left(1 + \eta_{\mathbb{S}}\boldsymbol{\phi}_t^\top\boldsymbol{G}_{\mathbb{S}}^{-1}\boldsymbol{\phi}_t\right)\right|. \tag{18}$$

The global volume quantifies the coverage of the temporal-pattern space as a whole. However, temporal dependence is often locally structured: retained windows with similar recent contexts can provide substantially stronger support for a query than patterns represented only at the global level. Global coverage alone may therefore fail to reveal whether a query has close counterparts. We consequently measure local support by the average KNN distance to the nearest windows:

$$v_t^{\text{local}} = \left(\frac{1}{k}\sum_{j\in\mathbb{J}_k(t;\mathbb{S})}\left\|\boldsymbol{\phi}_t - \boldsymbol{\phi}_j\right\|_2^2\right)^{1/2}. \tag{19}$$

We standardize the global and local scores and then combine them into a single SIS value:

$$s_t = \beta\text{Norm}_{\mathbb{S}}\left(v_t^{\text{global}}\right) + (1-\beta)\,\text{Norm}_{\mathbb{S}}\left(v_t^{\text{local}}\right), \qquad 0 \le \beta \le 1. \tag{20}$$

The coefficient controls the relative contribution of global non-redundancy and local isolation, and we use equal weights by default. A large SIS value indicates that the query is poorly supported by the strict retained set. Then, for each closure window, the conditional information of the diffusion model will be changed to:

$$\boldsymbol{c}_t = \left[\boldsymbol{x}_t, g\left(\boldsymbol{x}_t\right), s_t\right]. \tag{21}$$

At unlearning time, reverse diffusion generates residual samples $\left\{\hat{\tilde{\boldsymbol{r}}}_t^{(m)}\right\}_{m=1}^{M}$ from the conditional distribution $p_{\boldsymbol{\psi}}\left(\bar{\boldsymbol{r}} \mid \boldsymbol{c}_t\right)$, using the available information corresponding to the strictly reserved set. We average the generated residual samples and restore the original residual scale:

$$\widehat{R}_t^{D_r} = \text{mat}_{h,p}\left(\mu_r\mathbf{1} + \sigma_r\frac{1}{M}\sum_{m=1}^{M}\hat{\tilde{\boldsymbol{r}}}_t^{(m)}\right). \tag{22}$$

The final label for each affected window is then obtained by adding the generated residual correction to the retained-set NTK prediction:

$$\tilde{\boldsymbol{y}}_t = g_{\mathcal{D}_{\text{r}}}\left(\boldsymbol{x}_t\right) + \widehat{R}_t^{D_r}. \tag{23}$$

3.3 Label Field-Guided Unlearning

The diffusion model produces the label $\tilde{\boldsymbol{y}}_t$ for every closure window. Updating the model with such constructed targets is conceptually aligned with label-based and teacher-based unlearning strategies, which redirect the predictions of affected samples while preserving useful knowledge learned from the retained data (Chundawat et al., 2023; Kurmanji et al., 2023; Graves et al., 2021). In our setting, the labels are generated from retained-set information and are used to update the original model.

We initialize the unlearned model from the original parameters $\boldsymbol{\theta}_{\text{u}}^{(0)} = \boldsymbol{\theta}^*$. At each update step, we draw a shuffled mini-batch $\mathbb{M}_{\text{f}} \subset \mathcal{D}_{\text{f}}$ of closure queries and an independent random mini-batch $\mathbb{M}_{\text{r}} \subset \mathcal{D}_{\text{r}}$ of strictly retained supervised windows. The retained mini-batch provides an anchoring signal that stabilizes the update and preserves forecasting behaviour on unaffected temporal patterns and the closure mini-batch is aligned with the generated labels and supplies the forgetting signal that redirects predictions away from their original training-induced behaviour. We define the retained anchoring loss and the label alignment loss as

$$\mathcal{L}_{\text{r}} = \frac{1}{|\mathbb{M}_{\text{r}}|}\sum_{(\boldsymbol{x}_t,\boldsymbol{y}_t)\in\mathbb{M}_{\text{r}}}\left\|f_{\boldsymbol{\theta}_{\text{u}}}\left(\boldsymbol{x}_t\right) - \boldsymbol{y}_t\right\|_{\text{F}}^2,\ \mathcal{L}_{\text{f}} = \frac{1}{|\mathbb{M}_{\text{f}}|}\sum_{(\boldsymbol{x}_t,\boldsymbol{y}_t)\in\mathbb{M}_{\text{f}}}\left\|f_{\boldsymbol{\theta}_{\text{u}}}\left(\boldsymbol{x}_t\right) - \tilde{\boldsymbol{y}}_t\right\|_{\text{F}}^2. \tag{24}$$

The final update is computed according to the following objective:

$$\mathcal{L}_{\text{U}} = \lambda_{\text{r}}\mathcal{L}_{\text{r}} + \lambda_{\text{f}}\mathcal{L}_{\text{f}}. \tag{25}$$

4 EXPERIMENTS

4.1 DATASETS AND EXPERIMENTAL SETUPS

Datasets. We conduct experiments on four public time-series datasets. Air Quality (Chen, 2017) and Traffic(Lai et al., 2018) are used for univariate forecasting, whereas EEG(Zyma et al., 2019) and HAR70+(Logacjov & Ustad, 2023) are used for multivariate forecasting. For each dataset, we select 10 series or subjects and train a separate forecasting model for each one. Each model is evaluated under three deletion seeds, yielding 30 deletion requests per dataset. We first average the results across deletion seeds for each series or subject and then report the macro-average across the 10 series or subjects.

Evaluation Metrics. We evaluate unlearning effectiveness using membership indistinguishability (Shokri et al., 2017; Carlini et al., 2022) and functional agreement with exact retraining. The former is measured by the two-sided U/N AUC, where values closer to 0.5 indicate stronger forgetting. The latter is measured by the F-P Gap, namely the normalized RMSE between unlearned and exact-retrained predictions on forgetting inputs, reported as a percentage. Forecasting utility is evaluated using test NRMSE, while computational efficiency is measured by wall-clock time. Detailed metric and timing implementations are provided in the Appendix.

Baselines. We compare our model against a sizable collection of representative baselines: **SSD** (Foster et al., 2024), **NegGrad+** (Kurmanji et al., 2023), **Random label**, **Zero label**, **SCRUB-R** (Kurmanji et al., 2023), **TS Unlearn** (Wang et al., 2026), and we use the **Transformer** (Vaswani et al., 2017) as our base model. The details are described in the Appendix.

4.2 OVERALL PERFORMANCE

Table 1: Performance comparison between our method and other unlearning baselines on univariate and multivariate forecasting tasks. Unlearning effectiveness is evaluated by the two-sided U/N AUC and the F-P Gap(%), where U/N values closer to 0.5 indicate lower distinguishability between forgotten and never-seen samples, and a lower F-P Gap(%) indicates closer predictive agreement with exact retraining. Forecasting utility is evaluated by test NRMSE, with lower values indicating better predictive performance. Computational efficiency is measured by wall-clock time in seconds.

Method	Univariate Forecasting								Multivariate Forecasting							
	Air Quality				Traffic				EEG				HAR70			
	U/N	F-P Gap	NRMSE	Time	U/N	F-P Gap	NRMSE	Time	U/N	F-P Gap	NRMSE	Time	U/N	F-P Gap	NRMSE	Time
Original	0.6009	6.56	0.2546	–	0.6454	15.57	0.3941	–	0.6389	5.24	0.1069	–	0.6911	11.96	0.3495	–
Retrain	0.5034	–	0.2622	26.323	0.5291	–	0.3955	12.549	0.5235	–	0.1148	8.910	0.5210	–	0.3459	32.371
SSD	0.5218	52.8954	0.6683	0.946	0.5371	72.6908	0.8928	0.395	0.5180	54.1037	0.5877	0.227	0.5204	72.0281	0.5763	0.623
NegGrad+	0.5090	162.1498	1.5902	9.871	0.5412	140.9170	1.3989	4.340	0.5261	102.1933	0.9080	2.397	0.5372	288.0736	1.6082	10.051
Zero-label	0.5093	54.0268	0.6009	4.161	0.5387	51.8856	0.6896	2.150	0.5184	60.1355	0.5854	0.698	0.5227	58.6257	0.4214	1.009
Random-label	0.5069	54.2146	0.6052	0.775	0.5374	52.5161	0.7063	0.786	0.5183	60.0174	0.5846	0.695	0.5220	45.3235	0.5377	1.007
SCRUB-R	0.5195	8.4768	0.2720	6.854	0.5499	11.9866	0.4142	4.913	0.5368	18.9612	0.2149	2.992	0.5497	11.6701	0.3565	4.802
TS-Unlearn	0.5197	7.5967	0.2698	8.227	0.5458	13.5214	0.4189	3.579	0.5254	8.5701	0.1353	1.310	0.5385	12.8356	0.3539	1.933
Ours	**0.5001**	**4.6312**	**0.2600**	**11.710**	**0.5420**	**9.9251**	**0.3935**	**4.261**	**0.5244**	**3.7526**	**0.1110**	**2.322**	**0.5176**	**9.7703**	**0.3424**	**5.744**

Table 1 present the overall performance comparison of RDTU against all baselines across the four time-series datasets. ***Our RDTU consistently and significantly outperforms all competing methods under evaluation metrics of F-P Gap and NRMSE across all datasets.*** Specifically, RDTU achieves substantial improvements over the best baselines, SCRUB-R and TS-Unlearn, with relative gains ranging from 16.3% to 56.2% in F-P Gap and 3.2% to 18.0% in NRMSE. Although our performance on the U/N auc metric was not completely optimal, it can be observed that methods such as NegGrad+ and Random-label only achieved a slight advantage on U/N, but led to a complete deterioration in other metrics. This result is consistent with the limitations discussed earlier: aggressive gradient reversal or arbitrary target replacement may reduce membership distinguishability by broadly disrupting the learned forecasting function, rather than selectively removing the influence of the forgotten data. ***Compared to other forgetting models that did not experience training breakdown, we still maintained a good advantage on U/N auc, which indicates that we achieved excellent forgetting effects on these datasets.*** Extended experiments of different forecasting horizons shows in the Appendix.

4.3 Ablation Study

We conduct a systematic ablation study to investigate the contribution of each key component of our RDTU with the following ablated variants: **w/o diffusion** means we conduct the experiments without residual calculation; the **residual-MLP** and **residual-ARIMA** means we conduct the experiments with other methods such as MLP and ARIMA to get the residual; **w/o SIS** removes the SIS from the diffusion condition and generates residuals using only the NTK prediction. The results and comparison of these variants are shown in Table 2. ***Residual-related ablations show that removing residual correction degrades the quality of the constructed labels, confirming that the gap between the NTK predictor and the forecasting model cannot be ignored.*** Replacing diffusion with a residual MLP or ARIMA also yields inferior results, as deterministic regression may fail to capture the complex residual distribution. These results validate both residual correction and diffusion-based residual modeling. Removing SIS increases the F-P Gap by 3.1%-9.1%, confirming that structural information complements NTK predictions in residual generation.

Table 2: Ablation study of the key components of our method on different datasets.

Methods	Univariate Forecasting						Multivariate Forecasting					
	Air Quality			Traffic			EEG			HAR70		
	U/N	F-P Gap	NRMSE	U/N	F-P Gap	NRMSE	U/N	F-P Gap	NRMSE	U/N	F-P Gap	NRMSE
w/o Diffusion	0.5250	5.2378	0.2685	0.5455	11.3759	0.3946	0.5297	4.2684	0.1232	0.5190	9.8371	0.3434
Residual-MLP	0.5201	5.0807	0.2632	0.5454	10.3245	0.3940	0.5341	5.5752	0.2583	0.5261	10.064	0.3398
Residual-ARIMA	0.5392	4.9284	0.2609	0.5381	10.7915	0.3968	0.5218	4.2113	0.1135	0.5175	10.8921	0.3498
w/o SIS	0.5180	5.0545	0.2610	0.5475	10.2339	0.3964	0.5226	3.9427	0.1171	0.5213	10.0836	0.3452
Ours	**0.5001**	**4.6312**	**0.2600**	**0.5420**	**9.9251**	**0.3935**	**0.5244**	**3.7526**	**0.1110**	**0.5176**	**9.7703**	**0.3424**

4.4 Hyperparameter Analysis

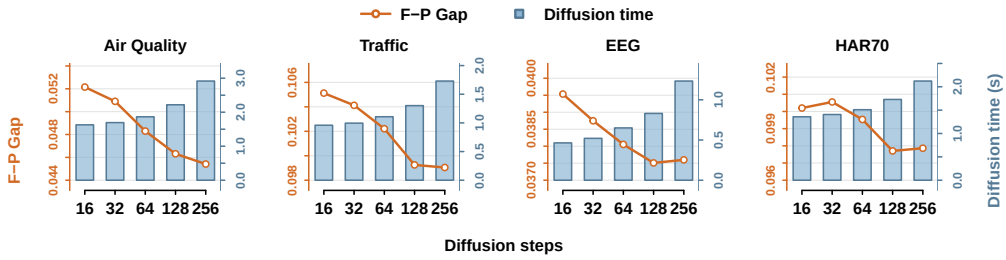


Figure 2: Sensitivity analysis of RDTU under different diffusion steps of Horizon Length=1.

We vary the number of diffusion steps while keeping all other settings fixed, and report the F-P Gap and diffusion-only runtime on different datasets. As shown in Figure 2, increasing the number of steps from 16 to 128 reduces the F-P Gap by 0.8% to 11.2%. Increasing the number of steps further to 256 results in only marginal changes of -0.01to 0.09 percentage points, while increasing the diffusion time by 0.39 to 0.70 seconds. ***These results indicate that insufficient denoising steps limit residual-generation quality, whereas excessive steps introduce additional cost with diminishing benefits.*** We therefore use $steps = 128$ as the default setting.

5 Conclusion

In this work, we introduced RDTU, a label-guided framework for approximate unlearning in time-series forecasting. RDTU provides an efficient prediction by constructing an NTK predictor, and the structural irrelevance score is used to characterize the global and local support conditions of each affected window. Conditional residual diffusion subsequently corrects the discrepancy between the NTK approximation and the corresponding neural predictor, and the resulting labels guide a lightweight unlearning update. Experiments on four univariate and multivariate forecasting benchmarks show that RDTU achieves the lowest F-P Gap and test NRMSE among the evaluated efficient unlearning methods, while maintaining low membership distinguishability and requiring substantially less computation than exact retraining.

AI USE STATEMENT

Generative AI tools were used solely to support the linguistic editing of this manuscript, including grammar correction, sentence rephrasing, readability improvement, and refinement of textual flow. They were not used to formulate research questions or hypotheses, develop the methodology, design or conduct experiments, analyze data, interpret results, or generate scientific conclusions. All AI-assisted text was critically reviewed and revised by the authors, who take full responsibility for the accuracy, originality, and integrity of the final manuscript.

ETHICS STATEMENT

This work investigates machine unlearning as a means of reducing the influence of designated observations on time-series forecasting models. All experiments use existing publicly available benchmark datasets; no new participants were recruited, and no new personal data were collected. The EEG and HAR datasets contain previously collected human-subject measurements, and this study uses only the publicly released research data. RDTU provides approximate rather than certified unlearning and should not be interpreted as offering a formal or legal guarantee of data erasure. Its effectiveness depends on the adopted threat model, evaluation protocol, and deployment setting. Applications in privacy-sensitive domains should therefore incorporate appropriate access controls, deletion audits, and regulatory review. Moreover, deletion requests and model updates should be documented and independently auditable to prevent unlearning techniques from being misused to obscure data provenance or selectively manipulate model behaviour.

REPRODUCIBILITY STATEMENT

To support reproducibility, all methods were tested using the same preprocessing, data partitioning, deletion requests, and evaluation procedures. The detailed experimental configurations and implementation methods are listed in the appendix. For each dataset, we evaluated 10 sequences or subjects under three deletion seeds, averaged the results for each sequence or subject under different seeds, and then reported the corresponding macro average. The source code, configuration files, selected sequence or subject identifiers, and random seeds will be included in the supplementary materials. The published content will include scripts for training the original model and re-training the precise model, as well as scripts for executing our experiments.

REFERENCES

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, 2015.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE symposium on security and privacy (SP)*, 2022.

Song Chen. Beijing Multi-Site Air Quality. UCI Machine Learning Repository, 2017.

Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *Advances in neural information processing systems*, 32, 2019.

Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Zonglin Di, Zhaowei Zhu, Jinghan Jia, Jiancheng Liu, Zafar Takhirov, Bo Jiang, Yuanshun Yao, Sijia Liu, and Yang Liu. Label smoothing improves machine unlearning. In *International Conference on Learning Representations*, 2026.

GDPR EU. General data protection regulation. *Official Journal of the European Union*, 2016.

Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Fast machine unlearning without retraining through selective synaptic dampening. In *Proceedings of the AAAI conference on artificial intelligence*, 2024.

Antonio Ginart, Melody Guan, Gregory Valiant, and James Zou. Making ai forget you: Data deletion in machine learning. In *Advances in Neural Information Processing Systems*, 2019.

Arda Goknil, Phu Nguyen, Sagar Sen, Dimitra Politaki, Harris Niavis, Karl John Pedersen, Abdillah Suyuthi, Abhilash Anand, and Amina Ziegenbein. A systematic review of data quality in cps and iot for industry 4.0. *ACM Computing Surveys*, 55:1–38, 2023.

Laura Graves, Vineel Nagisetty, and Vijay Ganesh. Amnesiac machine learning. In *Proceedings of the AAAI conference on artificial intelligence*, 2021.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, 2018.

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. In *Advances in Neural Information Processing Systems*, 2023.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2018.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems*, 2019.

Jaehoon Lee, Samuel Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein. Finite versus infinite neural networks: an empirical study. In *Advances in Neural Information Processing Systems*, 2020.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379: 20200209, 2021. doi: 10.1098/rsta.2020.0209.

Aleksej Logacjov and Astrid Ustad. HAR70+. UCI Machine Learning Repository, 2023.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary Chase Lipton, and J Zico Kolter. TOFU: A task of fictitious unlearning for LLMs. In *First Conference on Language Modeling*, 2024.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, 2017.

Hui Yie Teh, Andreas W Kempa-Liehr, and Kevin I-Kai Wang. Sensor data quality: A systematic review. *Journal of Big Data*, 7:11, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Ziyi Wang, Lixing Chen, Zhongqi Miao, Junhua Tang, Yang Bai, and Jianhua Li. Ts-unlearn: A dual-objective unlearning framework for time series forecasting. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2026.

Christopher Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. *Advances in neural information processing systems*, 13, 2000.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, 2021.

Greg Yang and Edward J Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In *International Conference on Machine Learning*, 2021.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. In *First Conference on Language Modeling*, 2024.

Tiago Zonta, Cristiano André Da Costa, Rodrigo da Rosa Righi, Miromar José De Lima, Eduardo Silveira Da Trindade, and Guann Pyng Li. Predictive maintenance in the industry 4.0: A systematic literature review. *Computers & industrial engineering*, 150:106889, 2020.

Igor Zyma, Sergii Tukaev, Ivan Seleznov, Ken Kiyono, Anton Popov, Mariia Chernykh, and Oleksii Shpenkov. Electroencephalograms during mental arithmetic task performance. *Data*, 4(1):14, 2019. doi: 10.3390/data4010014.

Appendix

A More Implementation Details

A.1 Datasets

We conduct experiments on four public time-series datasets. Air Quality and Traffic are used for univariate forecasting, whereas EEG and HAR70+ are used for multivariate forecasting. The forecasting horizon is one time step for all datasets. For each dataset, we select 10 series or subjects and train a separate forecasting model for each one. Each model is evaluated under three deletion seeds, yielding 30 deletion requests per dataset. We first average the results across deletion seeds for each series or subject and then report the macro-average across the 10 series or subjects.A deletion request is specified as a contiguous block of raw observations. We remove every forecasting example whose input or prediction target overlaps this block, thereby enforcing strict temporal deletion closure. Below, the input-window length denotes the number of historical observations provided to the forecasting model, while the deletion-block length denotes the number of consecutive raw observations selected for forgetting.

- *Air Quality*: This dataset contains hourly air-pollutant and meteorological measurements collected from 12 monitoring sites in Beijing between March 2013 and February 2017 (Chen, 2017). We treat the $PM_{2.5}$ measurements from each site as an independent univariate time series and predict its concentration at the next time step. We use an input window of 24 hourly observations.
- *Traffic*: This dataset contains hourly road occupancy measurements from 862 freeway sensors in the San Francisco Bay Area during 2015–2016 (Lai et al., 2018). We treat each sensor as an independent univariate time series and predict its occupancy rate at the next time step. We use an input window of 16 hourly observations.
- *EEG*: The EEGMAT dataset contains 19-channel EEG recordings from 36 subjects performing mental arithmetic tasks, sampled at 500 Hz (Zyma et al., 2019). We use the Fp1, Fp2, and F3 channels and jointly predict their values at the next time step. We use an input window of 32 samples.
- *HAR70+*: This dataset contains accelerometer recordings from 18 older adults wearing tri-axial sensors on the lower back and thigh, sampled at 50 Hz (Logacjov & Ustad, 2023). We use the `back_y`, `thigh_x`, and `thigh_y` channels and jointly predict their values at the next time step. We use an input window of 32 samples.

A.2 Phase-Specific SIS Construction

SIS is used in both diffusion training and counterfactual residual generation. Although the two phases share the same scoring framework, they evaluate opposite structural perturbations with respect to different reference sets. During diffusion training, the complete training set $\mathcal{D}$ serves as the reference, and SIS measures the structural change caused by removing the query. During residual generation, the strictly retained set $\mathcal{D}_{\mathrm{r}}$ becomes the reference, and SIS measures the change caused by adding the query. We represent these two operations using a phase-dependent reference set $\mathbb{S} \in \mathcal{D}, \mathcal{D}_{\mathrm{r}}$ and direction indicator.

$$\eta_{\mathbb{S}} = \begin{cases} -1, & \mathbb{S} = \mathcal{D}, \\ +1, & \mathbb{S} = \mathcal{D}_{\mathrm{r}}. \end{cases} \tag{26}$$

During diffusion training, $\mathbb{S} = \mathcal{D}$ and the global SIS component measures the structural change caused by removing the query from the complete training set. During residual generation, $\mathbb{S} = \mathcal{D}_{\mathrm{r}}$ and it measures the change caused by adding the query to the strictly retained set.

For each query window, let $\phi t = \boldsymbol{x}_t$ denote its vector representation, and let $\boldsymbol{\Phi}\mathbb{S}$ stack the representations of all windows in the reference set. The structural geometry of $\mathbb{S}$ is summarized by the following regularized second-moment matrix:

$$\boldsymbol{G}_{\mathbb{S}} = \boldsymbol{\Phi}_{\mathbb{S}}^{\top} \boldsymbol{\Phi}_{\mathbb{S}} + \lambda_{v,\mathbb{S}} \boldsymbol{I}_{d_x}, \qquad \lambda_{v,\mathbb{S}} = \epsilon_v \frac{\mathrm{Tr}\left(\boldsymbol{\Phi}_{\mathbb{S}}^{\top} \boldsymbol{\Phi}_{\mathbb{S}}\right)}{d_x}. \tag{27}$$

Here, d_x is the representation dimension, while the adaptive ridge coefficient $\lambda_{v,\mathbb{S}}$ stabilizes the matrix operations across reference sets with different scales. Based on this matrix, the global SIS component measures the absolute change in the log-determinant volume induced by the phase-specific perturbation:

$$v_{t,\mathbb{S}}^{\text{global}} = \frac{1}{2}\left|\log\det\left(\boldsymbol{G}_{\mathbb{S}} + \eta_{\mathbb{S}}\boldsymbol{\phi}_t\boldsymbol{\phi}_t^\top\right) - \log\det\left(\boldsymbol{G}_{\mathbb{S}}\right)\right|. \tag{28}$$

The global component captures overall directional change but does not indicate whether similar windows remain locally available. We therefore complement it with a nearest-neighbor measure of local support. To prevent the query from becoming its own nearest neighbor during diffusion training, the local reference set is defined as

$$\mathbb{S}_t^{\text{local}} = \begin{cases} \mathcal{D} \setminus \{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}, & \mathbb{S} = \mathcal{D}, \\ \mathcal{D}_{\text{r}}, & \mathbb{S} = \mathcal{D}_{\text{r}}. \end{cases} \tag{29}$$

Let $\mathbb{J}_k(t; \mathbb{S}_t^{\text{local}})$ denote the indices of the k nearest reference windows to query t. The local isolation score is then computed as

$$v_{t,\mathbb{S}}^{\text{local}} = \left(\frac{1}{k}\sum_{j\in\mathbb{J}_k(t;\mathbb{S}_t^{\text{local}})}\left\|\boldsymbol{\phi}_t - \boldsymbol{\phi}_j\right\|_2^2\right)^{1/2}. \tag{30}$$

The resulting phase-specific SIS is

$$s_t^{\mathbb{S}} = \beta\,\text{Norm}_{\mathbb{S}}\left(v_{t,\mathbb{S}}^{\text{global}}\right) + (1-\beta)\,\text{Norm}_{\mathbb{S}}\left(v_{t,\mathbb{S}}^{\text{local}}\right). \tag{31}$$

Therefore, $s_t^{\mathcal{D}}$ characterizes the structural role of the query in the complete training set and is used to train the diffusion model, whereas $s_t^{\mathcal{D}_{\text{r}}}$ characterizes its support under the retained-data condition and is used for counterfactual residual generation.

A.3 Conditional Residual Diffusion

RDTU does not directly apply diffusion to the forecasting targets. Instead, it models the discrepancy between a trained forecasting model and its corresponding NTK predictor. This design separates the counterfactual label into an NTK-based prediction and a residual correction. The NTK predictor provides the principal prediction under a given reference dataset, while the diffusion model captures the finite-width and nonlinear effects that cannot be represented by the NTK approximation.

Residual diffusion training. The diffusion model is trained under the complete-data condition. For each training window $\boldsymbol{x}_t$, we compute the source residual between the original forecasting model and the complete-data NTK predictor. Because residuals from different datasets and forecasting dimensions may have different scales, they are standardized before diffusion training:

$$\boldsymbol{R}_t^{\mathcal{D}} = f_{\boldsymbol{\theta}_{\mathcal{D}}^*}(\boldsymbol{x}_t) - g_{\mathcal{D}}(\boldsymbol{x}_t), \quad \bar{\boldsymbol{r}}_t^{\mathcal{D}} = \frac{\text{vec}\left(\boldsymbol{R}_t^{\mathcal{D}}\right) - \mu_r\mathbf{1}}{\sigma_r}. \tag{32}$$

Residual generation depends on both the model-based prediction and the structural role of the query in the reference dataset. We therefore condition the denoiser on the input window, the complete-data NTK prediction, and the complete-data SIS:

$$\boldsymbol{c}_t^{\mathcal{D}} = \left[\boldsymbol{x}_t^\top, g_{\mathcal{D}}(\boldsymbol{x}_t)^\top, s_t^{\mathcal{D}}\right]^\top. \tag{33}$$

Here, $g_{\mathcal{D}}(\boldsymbol{x}_t)$ represents the prediction induced by the complete training set, while $s_t^{\mathcal{D}}$ measures the structural change caused by removing the query from that set. The query window $\boldsymbol{x}_t$ retains its temporal context, allowing the diffusion model to distinguish residual patterns associated with different temporal inputs.

Let $\{\beta_s\}_{s=1}^{S_{\text{diff}}}$ denote the diffusion variance schedule, with $\alpha_s = 1-\beta_s$ and $\bar{\alpha}_s = \prod_{q=1}^{s}\alpha_q$. During training, a diffusion step s and Gaussian noise ϵ are sampled, the standardized source residual is

perturbed according to the forward diffusion process. The denoiser is trained to recover the injected noise:

$$\mathbf{r}_{t,s} = \sqrt{\bar{\alpha}_s}\tilde{\boldsymbol{r}}_t^{\mathcal{D}} + \sqrt{1-\bar{\alpha}_s}\epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \quad \mathcal{L}_{\text{diff}} = \mathbb{E}_{t,s,\epsilon}\left[\left\|\epsilon - \boldsymbol{\epsilon}_{\boldsymbol{\psi}}\left(\mathbf{r}_{t,s}, s, \boldsymbol{c}_t^{\mathcal{D}}\right)\right\|_2^2\right]. \tag{34}$$

Through this objective, the denoiser learns the conditional distribution of the residual discrepancy rather than a deterministic point estimate. This is particularly useful when similar NTK predictions correspond to different residual corrections because of nonlinear feature learning, stochastic optimization, or local temporal variation.

Transfer to the retained-data condition. At unlearning time, the trained denoiser is kept fixed. Because a deletion request typically affects only a limited portion of the complete training set, we assume that the residual relationship learned from the complete-data model remains transferable after replacing the complete-data condition with the retained-data condition. Importantly, the transferred object is the conditional residual pattern, rather than the original complete-data prediction.

At unlearning time, for each affected closure query, we replace the complete-data NTK prediction and SIS with their retained-set counterparts to construct the counterfactual condition:

$$\boldsymbol{c}_t^{\mathcal{D}_\mathrm{r}} = \left[\boldsymbol{x}_t^\top, g_{\mathcal{D}_\mathrm{r}}(\boldsymbol{x}_t)^\top, s_t^{\mathcal{D}_\mathrm{r}}\right]^\top. \tag{35}$$

The input window remains unchanged because it specifies the query for which the counterfactual label is required. In contrast, $g_{\mathcal{D}_\mathrm{r}}(\boldsymbol{x}_t)$ represents the prediction supported by the strictly retained set, and $s_t^{\mathcal{D}_\mathrm{r}}$ measures the structural change caused by adding the closure query to that set. The difference between $\boldsymbol{c}_t^{\mathcal{D}}$ and $\boldsymbol{c}_t^{\mathcal{D}_\mathrm{r}}$ therefore informs the diffusion model about the change from the original-data condition to the counterfactual retained-data condition.

Reverse residual generation. For each closure query, reverse diffusion begins with an independently sampled Gaussian vector $\hat{\tilde{\boldsymbol{r}}}_{t,S_{\text{diff}}}^{(m)}$. At each step, the denoiser predicts the noise under the retained-data condition and progressively reconstructs a standardized residual sample:

$$\hat{\tilde{\boldsymbol{r}}}_{t,S_{\text{diff}}}^{(m)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \quad \hat{\tilde{\boldsymbol{r}}}_{t,s-1}^{(m)} = \frac{1}{\sqrt{\alpha_s}}\left[\hat{\tilde{\boldsymbol{r}}}_{t,s}^{(m)} - \frac{\beta_s}{\sqrt{1-\bar{\alpha}_s}}\boldsymbol{\epsilon}_{\boldsymbol{\psi}}\left(\hat{\tilde{\boldsymbol{r}}}_{t,s}^{(m)}, s, \boldsymbol{c}_t^{\mathcal{D}_\mathrm{r}}\right)\right] + \sqrt{\tilde{\beta}_s}\boldsymbol{z}_s, \tag{36}$$

Here, $\boldsymbol{z}_s \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for $s > 1$, while $\boldsymbol{z}_1 = \mathbf{0}$ in the final denoising step. We repeat the reverse process M times and average the resulting samples to reduce sampling-specific variation. The averaged residual is then restored to its original scale and added to the retained-set NTK prediction:

$$\widehat{\boldsymbol{R}}_t^{\mathcal{D}_\mathrm{r}} = \mathrm{vec}^{-1}\left(\mu_r\mathbf{1} + \sigma_r\frac{1}{M}\sum_{m=1}^{M}\hat{\tilde{\boldsymbol{r}}}_{t,0}^{(m)}\right), \quad \tilde{\boldsymbol{y}}_t = g_{\mathcal{D}_\mathrm{r}}(\boldsymbol{x}_t) + \widehat{\boldsymbol{R}}_t^{\mathcal{D}_\mathrm{r}}. \tag{37}$$

The resulting $\tilde{\boldsymbol{y}}_t$ combines the retained-set prediction supplied by the NTK predictor with a residual correction generated under the corresponding retained structural condition. It is subsequently used as the guiding label for updating the original forecasting model on the strict temporal deletion closure.

A.4 End-to-End Unlearning Procedure

The complete procedure is summarized in Algorithm 1. Diffusion training can be performed once for the original model, while each deletion request only requires constructing the retained condition, generating the counterfactual labels, and updating the forecasting model.

A.5 Metrics

To evaluate *unlearning effectiveness*, we consider both membership indistinguishability and functional agreement with exact retraining. Membership indistinguishability is measured by the two-sided U/N AUC, defined as $\max\{a, 1-a\}$, where a is the ROC AUC for distinguishing forgotten samples (U) from never-seen samples (N). A value closer to 0.5 indicates that the forgotten samples are less distinguishable from non-members. Functional agreement is measured by the F-P Gap, defined as the RMSE between the predictions of the unlearned model and those of the exact-retraining

Algorithm 1 RDTU: Residual-Diffusion-Guided Time-Series Unlearning

Require: Original training set $\mathcal{D}$, original model $f_{\boldsymbol{\theta}^*_{\mathcal{D}}}$, and deletion request $\mathbb{T}_{\mathrm{f}}$
Ensure: Unlearned model $f_{\boldsymbol{\theta}_{\mathrm{u}}}$
Complete-data diffusion training
Construct the complete-data NTK predictor $g_{\mathcal{D}}$
Compute the source residuals $\boldsymbol{R}_t^{\mathcal{D}}$ and complete-data SIS values $s_t^{\mathcal{D}}$
Construct the conditions $\boldsymbol{c}_t^{\mathcal{D}}$ and train $\boldsymbol{\epsilon}_{\boldsymbol{\psi}}$ by minimizing $\mathcal{L}_{\mathrm{diff}}$
Counterfactual label generation
Construct the strict deletion closure $\mathcal{D}_{\mathrm{f}}(\mathbb{T}_{\mathrm{f}})$ and retained set $\mathcal{D}_{\mathrm{r}}(\mathbb{T}_{\mathrm{f}})$
Construct the retained-set NTK predictor $g_{\mathcal{D}_{\mathrm{r}}}$
for each $(\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathcal{D}_{\mathrm{f}}$ **do**
    Compute the retained-set SIS $s_t^{\mathcal{D}_{\mathrm{r}}}$
    Generate $\widehat{\boldsymbol{R}}_t^{\mathcal{D}_{\mathrm{r}}}$ from $p_{\boldsymbol{\psi}}(\bar{\boldsymbol{r}} \mid \boldsymbol{c}_t^{\mathcal{D}_{\mathrm{r}}})$
    Construct $\tilde{\boldsymbol{y}}_t \leftarrow g_{\mathcal{D}_{\mathrm{r}}}(\boldsymbol{x}_t) + \widehat{\boldsymbol{R}}_t^{\mathcal{D}_{\mathrm{r}}}$
end for
Label field-guided unlearning
Initialize $\boldsymbol{\theta}_{\mathrm{u}}^{(0)} \leftarrow \boldsymbol{\theta}^*_{\mathcal{D}}$
Optimize $\boldsymbol{\theta}_{\mathrm{u}}$ on retained targets and generatedclosure labels using $\lambda_{\mathrm{retain}}\mathcal{L}_{\mathrm{r}} + \lambda_{\mathrm{label}}\mathcal{L}_{\mathrm{f}}$
return $f_{\boldsymbol{\theta}_{\mathrm{u}}}$

model on the forgetting inputs, normalized by the standard deviation of the raw training targets and reported as a percentage. A lower F-P Gap indicates closer agreement with exact retraining. To evaluate *forecasting utility*, we report the test NRMSE, computed as the test RMSE divided by the standard deviation of the raw training series before deletion, where a lower value indicates better predictive performance. *Computational efficiency* is measured by the wall-clock time required to complete each unlearning method. For our method, the reported time includes NTK computation, diffusion training and sampling, and model updating.

Membership Inference Attack We conduct the membership audit at the level of temporal deletion units rather than isolated observations. An audit candidate may therefore contain multiple forecasting windows induced by the same temporal interval. Following the strict temporal deletion closure defined in Equation 2, we define the closure operator for an arbitrary temporal interval $\mathbb{T}'$ as

$$\mathcal{C}\left(\mathbb{T}'\right) = \left\{(\boldsymbol{x}_t, \boldsymbol{y}_t) \mid \mathbb{I}_t \cap \mathbb{T}' \neq \emptyset\right\}, \qquad \mathbb{I}_t = [t-\ell, t+h). \tag{38}$$

Let $\mathbb{T}_{\mathrm{f}}$ denote the raw timestamps included in a deletion request, and let $\mathbb{T}_{\mathrm{n}}$ denote an equally sized set of timestamps held out before the original model is trained. The forgotten and never-seen audit collections are defined as

$$\mathcal{A}_{\mathrm{U}} = \mathcal{C}\left(\mathbb{T}_{\mathrm{f}}\right), \qquad \mathcal{A}_{\mathrm{N}} = \mathcal{C}\left(\mathbb{T}_{\mathrm{n}}\right). \tag{39}$$

Thus, $\mathcal{A}_{\mathrm{U}}$ contains every forecasting window whose input context or prediction target overlaps the requested deletion interval. These windows were used to train the original model $f_{\boldsymbol{\theta}^*_{\mathcal{D}}}$ but should behave as non-members after unlearning. In contrast, $\mathcal{A}_{\mathrm{N}}$ is excluded before the original model is trained and therefore consists of genuine non-members.

The never-seen intervals are fixed for each user across deletion seeds, whereas the forgotten intervals are independently sampled for each deletion request. We additionally select a retained-member audit collection $\mathcal{A}_{\mathrm{M}} \subset \mathcal{D}_{\mathrm{r}}$. To reduce distributional bias, candidates in $\mathcal{A}_{\mathrm{M}}$ and $\mathcal{A}_{\mathrm{N}}$ are matched to those in $\mathcal{A}_{\mathrm{U}}$ using temporal descriptors and Hungarian matching. The resulting M/U/N groups are mutually disjoint and contain the same number of temporal audit units. For multivariate data, all variables observed within the same temporal window are jointly treated as one forecasting sample rather than as independent membership candidates.

Candidate-conditioned LiRA We implement membership inference using candidate-conditioned LiRA. For each temporal audit unit i, shadow models are trained using the same background training data while alternately including or excluding that candidate. Because one candidate may induce multiple closure windows, the forecasting statistics over all its associated windows are aggregated into a single attack-feature vector $\boldsymbol{z}_i(f)$. The feature vector contains squared error, absolute error, signed error, residual variation, prediction statistics, and target statistics.

Using the outputs of the shadow models, we fit separate diagonal Gaussian distributions to the observations obtained when candidate i is included in or excluded from shadow-model training:

$$p_{c,i} = \mathcal{N}\left(\boldsymbol{\mu}_i^c, \operatorname{diag}\left((\boldsymbol{\sigma}_i^c)^2\right)\right), \qquad c \in \{\text{in}, \text{out}\}. \tag{40}$$

For a target forecasting model $f_{\boldsymbol{\theta}}$, the candidate-conditioned LiRA score is the log-likelihood ratio

$$s_i^{\text{LiRA}}(f_{\boldsymbol{\theta}}) = \log p_{\text{in},i}\left(\boldsymbol{z}_i(f_{\boldsymbol{\theta}})\right) - \log p_{\text{out},i}\left(\boldsymbol{z}_i(f_{\boldsymbol{\theta}})\right). \tag{41}$$

A larger score indicates that the behavior of candidate i is more consistent with that of a training member. When auditing the original model $f_{\boldsymbol{\theta}^*_{\mathcal{D}}}$, both $\mathcal{A}_{\text{M}}$ and $\mathcal{A}_{\text{U}}$ are treated as members, whereas $\mathcal{A}_{\text{N}}$ is treated as the non-member group. When auditing the unlearned model $f_{\boldsymbol{\theta}_{\text{u}}}$, only $\mathcal{A}_{\text{M}}$ retain member-like behavior, while candidates in both $\mathcal{A}_{\text{U}}$ and $\mathcal{A}_{\text{N}}$ should exhibit non-member behavior. We use separate shadow-model distribution fits for the original-model and unlearning audits.

Two-sided U/N AUC. Let $a_{\text{U/N}}$ denote the conventional ROC AUC obtained by distinguishing the LiRA scores of forgotten candidates in $\mathcal{A}_{\text{U}}$ from those of never-seen candidates in $\mathcal{A}_{\text{N}}$. Because the direction of the LiRA score may be reversed after an unlearning update, a conventional AUC below 0.5 does not necessarily imply that the two groups are indistinguishable. We therefore report the two-sided U/N AUC:

$$\text{AUC}_{\text{U/N}}^{\text{two}} = \max\left\{a_{\text{U/N}}, 1 - a_{\text{U/N}}\right\}. \tag{42}$$

The resulting metric lies in $[0.5, 1]$. A value close to 0.5 indicates that the forgotten and never-seen candidates are difficult to distinguish, whereas a larger value indicates that they remain separable regardless of the score direction.

RNMSE Forecasting performance is evaluated after restoring both predictions and targets to their original data scale. Let $\mathcal{Q} = \{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}_{t \in \mathbb{T}_{\mathcal{Q}}}$ denote an evaluation set, where $\mathbb{T}_{\mathcal{Q}}$ is its set of forecast origins. For a forecasting model $f_{\boldsymbol{\theta}}$, its root mean squared error on $\mathcal{Q}$ is defined as

$$\text{RMSE}_{\mathcal{Q}}(f_{\boldsymbol{\theta}}) = \sqrt{\frac{1}{|\mathbb{T}_{\mathcal{Q}}|\, hp} \sum_{t \in \mathbb{T}_{\mathcal{Q}}} \left\| f_{\boldsymbol{\theta}}(\boldsymbol{x}_t) - \boldsymbol{y}_t \right\|_{\text{F}}^2}. \tag{43}$$

Here, h denotes the forecasting horizon and p denotes the number of variables. For the univariate setting considered in our notation, $p = 1$; for multivariate forecasting, the prediction and target of each window contain hp scalar values.

Let σ_{train} denote the standard deviation of the raw training values used to train the original model, computed before applying the deletion request. We normalize the forecasting error as

$$\text{NRMSE}_{\mathcal{Q}}(f_{\boldsymbol{\theta}}) = \frac{\text{RMSE}_{\mathcal{Q}}(f_{\boldsymbol{\theta}})}{\sigma_{\text{train}}}. \tag{44}$$

The test NRMSE is obtained by setting $\mathcal{Q} = \mathcal{D}_{\text{test}}$. A lower NRMSE indicates better forecasting performance relative to the variation of the original training series.

F-P Gap To measure the functional agreement between an evaluated model f and exact retraining on the windows affected by the deletion request, we define the Forgetting-set Prediction Gap as

$$\text{F-P Gap}(f) = \frac{100\%}{\sigma_{\text{train}}} \sqrt{\frac{1}{|\mathcal{D}_{\text{f}}| hp} \sum_{(\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathcal{D}_{\text{f}}} \left\| f(\boldsymbol{x}_t) - f_{\boldsymbol{\theta}^*_{\mathcal{D}_{\text{r}}}}(\boldsymbol{x}_t) \right\|_{\text{F}}^2}. \tag{45}$$

Here, $f_{\boldsymbol{\theta}^*_{\mathcal{D}_{\text{r}}}}$ is the model trained from scratch on the retained set $\mathcal{D}_{\text{r}}$, σ_{train} is the standard deviation of the raw training targets before deletion, h is the forecasting horizon, and p is the number of variables. The factor 100 expresses the normalized gap as a percentage. Unlike forecasting NRMSE, which compares predictions with the ground-truth, the F-P Gap compares the predictions of f with those of exact retraining on the forgetting set $\mathcal{D}_{\text{f}}$. A lower F-P Gap indicates closer functional agreement with exact retraining, and a value of zero indicates identical predictions on all forgetting inputs.

A.6 BASELINES

We compare RDTU with the following representative unlearning baselines. **Original** denotes the forecasting model before unlearning, while **Retrain** trains the same architecture from scratch on the strictly retained set and serves as the exact-unlearning reference. **SSD** (Foster et al., 2024) uses Fisher information to identify parameters disproportionately associated with the forgetting set and selectively dampens them. **NegGrad+** (Kurmanji et al., 2023) combines gradient ascent on the forgetting set with gradient descent on the retained set. **Zero-label** replaces the targets of affected windows with zero vectors, whereas **Random-label** assigns randomly generated continuous targets following the random-labeling strategy. **SCRUB-R** (Kurmanji et al., 2023) adopts a teacher–student framework that moves the student away from the original model on the forgetting set while preserving its predictions on the retained set. **TS-Unlearn** (Wang et al., 2026) constructs replacement targets by feeding Gaussian noise into the original model and dynamically balances the forgetting and retention objectives through Pareto optimization.

B MORE EXPERIMENTS

To demonstrate the robustness and applicability of our model, we conducted additional expansion experiments with different forecasting horizon lengths. Table 3 and Table 4 respectively show the experimental results for horizon lengths of 3.

Table 3: Robustness comparison on univariate datasets under a forecasting horizon of $H = 3$. U/N values closer to 0.5 indicate better unlearning effectiveness, while lower F-P Gap, NRMSE, and runtime indicate better performance.

Method	**Air Quality**				**Traffic**			
	U/N	F-P Gap	NRMSE	Time	U/N	F-P Gap	NRMSE	Time
Original	0.6082	10.91	0.4084	–	0.6716	15.57	0.4908	–
Retrain	0.5400	–	0.4131	20.480	0.5478	–	0.5063	12.640
SSD	0.5429	43.31	0.6725	0.77	0.5501	79.35	0.9927	0.450
NegGrad+	0.5356	167.58	1.6929	7.88	0.5346	79.90	0.8708	3.930
Zero-label	0.5348	50.79	0.6918	3.33	0.5507	49.89	0.7294	1.820
Random-label	0.5367	50.51	0.6933	3.74	0.5490	50.05	0.7425	2.170
SCRUB-R	0.5439	10.21	0.4077	8.43	0.5627	14.87	0.4995	4.400
TS-Unlearn	0.5415	10.74	0.4128	6.43	0.5464	17.64	0.5077	3.780
Ours	0.5483	8.01	0.3977	10.95	0.5377	14.75	0.4970	4.550

The results show that RDTU remains effective when the forecasting horizon is increased to 3. In particular, RDTU achieves the lowest F-P Gap among all approximate-unlearning methods on all four datasets. Compared with the strongest baseline for each dataset, RDTU reduces the F-P Gap by 21.5%, 0.8%, 19.6%, and 13.2% on Air Quality, Traffic, EEG, and HAR70, respectively. The improvement is particularly pronounced on Air Quality and EEG, indicating that the counterfactual labels generated by RDTU remain close to the predictions of exact retraining even when each prediction contains multiple future steps.

RDTU also obtains the lowest test NRMSE among the approximate-unlearning methods across all datasets, reducing NRMSE by approximately 0.2%–2.5% relative to the best-performing baseline. Its U/N AUC values remain between 0.5207 and 0.5483, demonstrating competitive membership indistinguishability without the severe forecasting degradation observed for SSD, NegGrad+, Zero-label, and Random-label. For example, some of these methods obtain U/N AUC values close to 0.5, but simultaneously produce F-P Gaps above 40% and substantially larger NRMSE values. ***This suggests that their apparent membership indistinguishability is partly caused by excessive disruption of the forecasting function rather than close approximation to exact retraining.***

Table 4: Robustness comparison on multivariate datasets under a forecasting horizon of $H = 3$. U/N values closer to 0.5 indicate better unlearning effectiveness, while lower F-P Gap, NRMSE, and runtime indicate better performance.

Method	**EEG**				**HAR70**			
	U/N	F-P Gap	NRMSE	Time	U/N	F-P Gap	NRMSE	Time
Original	0.6759	5.24	0.1986	–	0.6946	12.53	0.3977	–
Retrain	0.5152	–	0.2055	8.980	0.5162	–	0.3975	30.71
SSD	0.5257	59.03	0.6593	0.230	0.5247	82.05	0.6331	0.55
NegGrad+	0.5267	95.88	0.8748	2.350	0.5314	208.95	1.1839	9.28
Zero-label	0.5244	60.22	0.6029	0.360	0.5231	58.71	0.4528	0.62
Random-label	0.5247	60.02	0.6012	0.660	0.5235	45.24	0.5611	1.00
SCRUB-R	0.5452	8.51	0.2191	2.810	0.5613	12.23	0.3976	9.92
TS-Unlearn	0.5324	8.31	0.2155	0.930	0.5477	12.75	0.4016	1.49
Ours	0.5207	6.68	0.2107	2.670	0.5305	10.61	0.3968	7.61

We also conducted experiments with a horizon length of 6. The results of these experiments are presented in Tables 5 and Tables 6. The results show that in this configuration, our model also achieved excellent results.

Table 5: Robustness comparison on univariate datasets under a forecasting horizon of $H = 6$. U/N values closer to 0.5 indicate better unlearning effectiveness, while lower F-P Gap, NRMSE, and runtime indicate better performance.

Method	**Air Quality**				**Traffic**			
	U/N	F-P Gap	NRMSE	Time	U/N	F-P Gap	NRMSE	Time
Original	0.6653	12.39	0.5379	–	0.6645	17.02	0.5821	–
Retrain	0.5330	–	0.5556	16.54	0.5698	–	0.5916	10.94
SSD	0.5312	51.25	0.7903	1.02	0.5661	78.50	0.9965	0.76
NegGrad+	0.5423	105.31	1.1596	7.48	0.5583	53.88	0.7481	3.53
Zero-label	0.5310	50.57	0.7553	4.14	0.5723	49.50	0.7698	2.84
Random-label	0.5304	49.75	0.7516	3.92	0.5745	49.46	0.7805	2.47
SCRUB-R	0.5360	46.37	0.6910	5.98	0.5732	51.38	0.8780	3.16
TS-Unlearn	0.5562	11.69	0.5494	8.01	0.5743	30.89	0.6613	4.99
Ours	0.5521	9.95	0.5450	12.44	0.5698	16.09	0.5803	7.54

C More Hyperparameter Analysis

C.1 Influence of Diffusion Steps on Different Horizon Length

The same with the setting where the horizon length is 1, we also examined the impact of the diffusion step number on the model with a horizon length of 3. Figure 3 presents the detailed experimental results. ***From the figure, we can observe that the trend of the influence of the diffusion step count on the model's performance and training duration is consistent with the case where the horizon is 1.*** Therefore, we have set the diffusion step count here to 128 as well.

C.2 Influence of Forget Loss Weight

Figure 4 examines the effect of the loss-weight ratio $\lambda_{\mathrm{f}}/\lambda_{\mathrm{r}}$. We select the ratio primarily according to the F-P Gap and NRMSE, because it directly measures the functional agreement between the

Table 6: Robustness comparison on multivariate datasets under a forecasting horizon of $H = 6$. U/N values closer to 0.5 indicate better unlearning effectiveness, while lower F-P Gap, NRMSE, and runtime indicate better performance.

Method	**EEG**				**HAR70**			
	U/N	F-P Gap	NRMSE	Time	U/N	F-P Gap	NRMSE	Time
Original	0.7301	10.74	0.3678	–	0.6982	13.88	0.4393	–
Retrain	0.5210	–	0.3770	–	0.5215	–	0.4430	34.45
SSD	0.5268	58.46	0.7230	0.62	0.5187	81.82	0.6467	0.86
NegGrad+	0.5305	93.27	0.9043	2.60	0.5271	182.69	0.9548	9.91
Zero-label	0.5289	56.45	0.6505	0.79	0.5199	59.14	0.4808	1.10
Random-label	0.5284	56.31	0.6489	0.36	0.5241	45.48	0.5931	0.67
SCRUB-R	0.5653	13.49	0.3872	2.83	0.5596	13.50	0.4407	4.93
TS-Unlearn	0.5458	14.57	0.3883	1.20	0.5460	14.57	0.4431	1.78
Ours	0.5348	11.02	0.3792	12.97	0.5361	11.26	0.4379	8.23

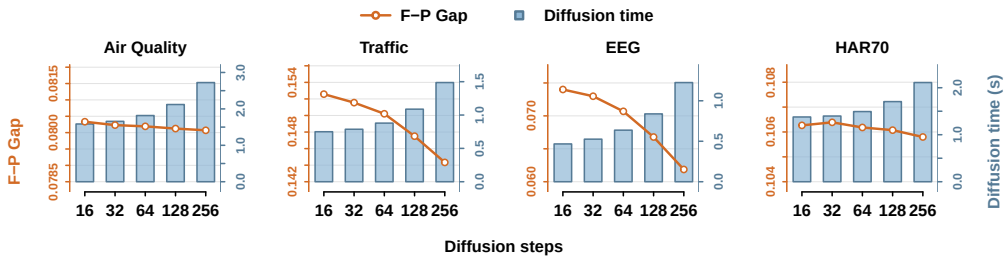


Figure 3: Sensitivity analysis of RDTU under different diffusion steps of Horizon Length=3. The curves report the F-P Gap and the time cost of diffusion.

unlearned model and the exact-retrained model on the affected windows and retain set. A lower F-P Gap therefore indicates that the generated labels guide the model more closely toward the desired post-deletion behavior, and a lower RNMSE represents the model maintained good performance. Based on this criterion, the best ratios are 1.8, 0.8, 0.2, and 0.3 for Air Quality, Traffic, EEG, and HAR70, respectively. Overall, the corresponding test NRMSE changes remain relatively moderate, indicating that optimizing the F-P Gap does not cause substantial degradation in forecasting utility. ***These results also show that the preferred balance between the forgetting loss and the retained anchoring loss varies across datasets because of their different temporal structures and prediction scales.***

D Future Work

D.1 Computational scalability

Although RDTU is more efficient than exact retraining, its computational cost may increase for long sequences, high-dimensional multivariate data, and large forecasting architectures. In particular, empirical NTK construction requires computing and storing parameter-gradient features for numerous forecasting windows, while residual generation involves multiple denoising steps for every affected query. Future work could reduce these costs through incremental Nystr"om approximation, low-rank kernel representations, and cached NTK features that can be reused across deletion requests. The retained-reference statistics required by SIS could likewise be updated incrementally instead of being recomputed after each deletion. On the diffusion side, accelerated or distilled samplers, batched residual generation, and adaptive denoising schedules could further reduce inference time. For example, queries with strong retained support could use fewer denoising steps, whereas struc-

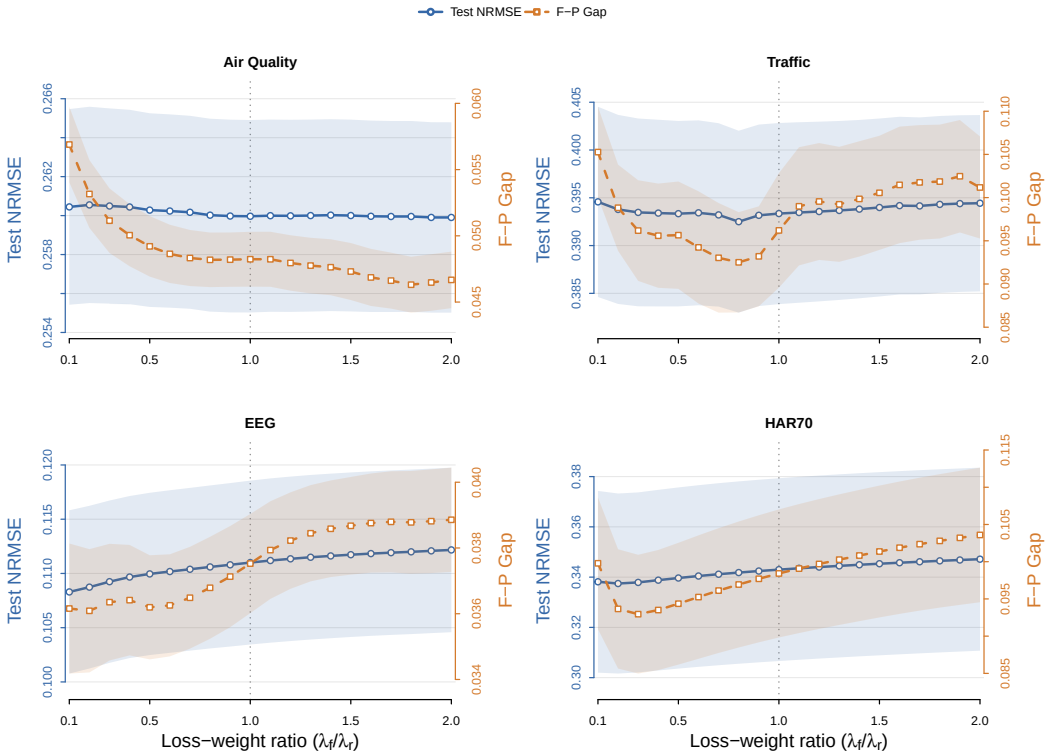


Figure 4: Sensitivity analysis of RDTU under different loss-weight ratios $\lambda_{\mathrm{f}}/\lambda_{\mathrm{r}}$ of Horizon Length=1. The curves report the mean test NRMSE and F-P Gap, while the shaded regions indicate the corresponding standard errors over 30 runs. Lower values indicate better performance for both metrics.

turally irreplaceable queries with high SIS values could receive additional computation. A comprehensive scalability study should separately report one-time preprocessing costs, per-request latency, peak memory consumption, and amortized runtime as the number of windows, variables, deletion requests, and diffusion steps increases.

D.2 Broader Evaluation and Stronger Guarantees

Future evaluations should examine RDTU under a broader range of forecasting and deletion conditions. Relevant settings include longer forecasting horizons, higher-dimensional multivariate series, irregular sampling, missing observations, temporal distribution shifts, and deletion intervals spanning rare or rapidly changing patterns. It would also be valuable to vary the deletion ratio, interval length, and number of sequential requests to determine when the residual-transfer assumption begins to deteriorate. Evaluations with recurrent networks, temporal convolutional networks, Transformers, and state-space models would further assess the generality of the architecture-agnostic design. In addition, the current evaluation measures empirical agreement with exact retraining and resistance to membership inference, but these results do not provide a formal guarantee of data removal. Future studies should consider adaptive black-box and white-box attacks that exploit model outputs, gradients, parameters, or intermediate representations, together with reconstruction and model-inversion attacks. A complementary theoretical direction is to bound the discrepancy between the unlearned and exact-retrained models in terms of the NTK approximation error, residual-generation error, and optimization error. Such analyses could support verifiable stopping criteria or certified variants of RDTU for deployment in privacy-sensitive applications.