DIAMOND: DIFFUSION-INFORMED PATCH MASKING WITH COUNTERFACTUAL AUDITING FOR TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Masking strategies have significantly advanced time-series forecasting by suppressing redundant historical information while preserving predictive patterns. Despite the progress, the performance of existing approaches is still far from satisfactory due to inflexible masking rates across time series and the risk of masking crucial signals. To address these limitations, we propose a novel approach named Diffusion-Informed Patch Masking with Counterfactual Auditing (DIAMOND) for time series forecasting. The core of our DIAMOND is to identify optimal masking patterns tailored to individual time series from two complementary perspectives: *diffusion-based reconstruction* and *counterfactual auditing*. In particular, we first introduce a conditional diffusion model to reconstruct each time series, where patches exhibiting high reconstruction losses are assigned a redundancy prior for masking guidance. Then, we introduce two disentangled predictors to separately determine masking ratios and masking patterns for individual time series. More importantly, we construct counterfactual samples by recovering individual masked patches and track the variation in prediction losses to prevent patches with predictive patterns from being masked. Extensive experiments on benchmark datasets validate the superiority of the proposed DIAMOND in comparison with a range of state-of-the-art approaches. Source code is available at https://anonymous.4open.science/r/Diamond-code-A812.

1 INTRODUCTION

Time-series forecasting is a fundamental task that supports data-driven decision-making in energy management, traffic monitoring, financial analysis, and industrial operations (Zhou et al., 2021; Wen et al., 2022). Accurate forecasts enable effective resource allocation, operational planning, and risk management, making it essential to extract predictive patterns from increasingly long and complex historical observations. In recent years, deep forecasting models based on convolutional neural networks (Wu et al., 2023), multilayer perceptrons (Wang et al., 2024), and transformer architectures (Liu et al., 2024b) have improved forecasting performance by modeling complex temporal dependencies. Among them, patch-based methods encode consecutive observations into compact temporal tokens, reducing sequence length while preserving local dynamics (Liu et al., 2025). Nevertheless, extending the historical context does not guarantee more useful information. Redundant fluctuations, task-irrelevant disturbances, and uninformative segments may dilute predictive signals, induce spurious dependencies, and ultimately impair forecasting generalization (Qiu et al., 2025).

To mitigate information redundancy in historical sequences, masking has been widely adopted for time-series representation learning and forecasting regularization. Early masked-modeling approaches mainly rely on random corruption. TST (Zerveas et al., 2021) randomly masks a subset of time points and reconstructs them to learn general-purpose temporal representations, whereas PatchTST (Nie et al., 2022) extends the masking unit to contiguous temporal patches. However, these methods use predefined masking ratios and random positions, ignoring variations in sample redundancy and temporal relevance. Recent studies have introduced adaptation at different levels. SCAM (Yang et al., 2026) replaces potentially overfitted target components with reconstruction-derived pseudo-labels, DropoutTS (Zhong et al., 2026) maps sample-level spectral noise estimates to adaptive feature-dropout rates, and AMRC (Liang et al., 2026) evaluates sampled temporal-prefix

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.

masks using forecasting loss and regularizes the model. However, these methods provide only partial control over input information removal, as the masking amount and temporal locations are not jointly determined based on their effects on downstream forecasting.

In this work, we study task-aligned adaptive patch masking for time-series forecasting. This raises two coupled challenges. ***I) How can the masking amount adapt to each sample?*** A fixed mask ratio applies the same degree of information removal to histories with different temporal structures. Yet patches in one sample may be readily recovered from the remaining context, whereas those in another may be difficult to recover. Consequently, the same ratio may retain substantial recoverable content in one sample while removing more distinctive content from another. Conditional recoverability offers a sample-level signal for adjusting the relative masking amount, although it does not directly determine a forecasting-loss-constrained budget. ***II) How can patch selection reflect forecasting impact?*** Recoverability alone does not establish predictive irrelevance. A patch reconstructed accurately from its neighbors may still contain patterns useful for forecasting the future. Moreover, the loss of a fully masked sequence reflects the combined effects of multiple removed patches and cannot isolate the contribution of any one patch. Patch prioritization therefore requires local, task-aligned evidence, obtained by comparing forecasting losses before and after restoring an individual patch under the same mask context. These challenges motivate a closer examination of how the amount and placement of input masking can be guided by different signals.

To address the aforementioned challenges, we propose a novel approach named Diffusion-Informed Patch Masking with Counterfactual Auditing (`DIAMOND`) for time series forecasting. The goal of `DIAMOND` is to adapt the amount of training-time masking using conditional recoverability cues and to prioritize patches using their observed counterfactual effects on forecasting loss. Specifically, given a historical sequence, `DIAMOND` first employs a conditional diffusion branch to reconstruct masked patches from visible context. The resulting patch-wise reconstruction errors provide recoverability cues for mask generation. On this basis, we introduce a hierarchical controller in which a Scalar Head predicts the masking ratio for each sample, while a Direction Head assigns masking priority to individual patches. Their outputs are combined through budget-constrained Top-k selection to form the training mask. Moreover, while a patch may be easily reconstructed from its context, removing it can still impair forecasting. Consequently, `DIAMOND` periodically restores masked patches one at a time and measures the resulting changes in prediction loss. These counterfactual observations refine patch priorities. Finally, a shared encoder supports joint forecasting and diffusion reconstruction during training. Extensive experiments across forecasting benchmarks and backbones demonstrate the effectiveness of `DIAMOND`.

In summary, we briefly outline the main contributions of this paper as follows: ❶ ***New Perspective:*** We distinguish conditional recoverability from predictive relevance and view adaptive patch masking as two coupled decisions. This perspective motivates separate evidence for guiding sample-level mask ratios and assessing patch-level forecasting effects. ❷ ***Novel Framework:*** We develop a hierarchical mask controller that decomposes adaptive masking into sample-level budget estimation and patch-level safety ranking. Conditional diffusion yields patch-level training signals to guide relative masking amounts, while paired restoration provides task-based supervision for patch ranking. ❸ ***Extensive Experiments:*** Extensive experiments across multiple time-series benchmarks and backbones demonstrate the effectiveness and generalizability of `DIAMOND`.

2 Related Work

Masking Strategies for Time-Series Forecasting. Masking strategies for time-series forecasting have gradually evolved from random perturbation to prediction-oriented information selection (Dong et al., 2023; 2024). TST (Zerveas et al., 2021) established an early masked reconstruction paradigm by randomly masking individual observations and pretraining a Transformer to recover them. PatchTST (Nie et al., 2022) extended the masking unit from individual time points to temporal patches, thereby preserving local temporal patterns within each token. MOMENT (Goswami et al., 2024) further scaled masked-patch reconstruction to large and heterogeneous time-series corpora. Moving beyond deterministic reconstruction, TSDE (Senane et al., 2024) developed a unified masking mechanism for imputation, interpolation, and forecasting, and employed a conditional diffusion process to predict additive noise within masked regions. From an efficiency perspective, DropPatch (Qiu et al., 2025) directly removes randomly selected patch tokens. More recently, mask-

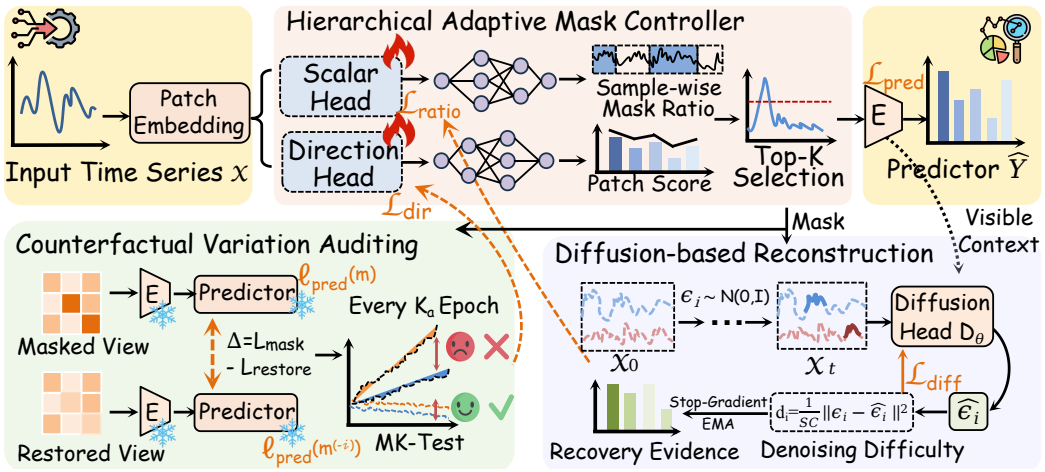


Figure 1: Overview of DIAMOND. Diffusion-based recoverability informs sample mask-ratio estimation, while periodic counterfactual supervision refines patch maskability scores. The ratios and scores are combined via Top-k selection to construct adaptive training masks.

ing strategies have begun to incorporate forecasting objectives into their decisions. AMRC (Liang et al., 2026) makes masking predictions by sampling different historical-prefix truncations.

Diffusion Models for Time-Series Reconstruction. Diffusion models provide a natural mechanism for modeling complex temporal distributions through progressive corruption and reverse denoising (Shen & Kwok, 2023; Liu et al., 2024a). TimeGrad (Rasul et al., 2021) combines an autoregressive recurrent network with denoising diffusion to generate correlated multivariate forecasts. D3VAE (Li et al., 2022) further integrates coupled diffusion, multiscale denoising, and latent disentanglement into generative forecasting. Subsequent studies have shifted from task-specific generation toward task-flexible and structure-aware denoising (Zhao et al., 2026). TSDiff (Kollovieh et al., 2023) trains a task-agnostic unconditional diffusion model and employs self-guidance for forecasting, refinement, and synthesis. MG-TSD (Fan et al., 2024) introduces multi-granularity targets to provide structural guidance at different diffusion stages. Recent studies further adapt diffusion models to the statistical properties and local structures of time series. NsDiff (Ye et al., 2025) models time-varying uncertainty through location noise and uncertainty-aware scheduling. However, existing diffusion methods mainly optimize denoising and reconstruction, without calibrating patch recoverability against its actual effect on forecasting loss (Solano Carrillo et al., 2026). DIAMOND performs this calibration through counterfactual task supervision and jointly learns sample-specific masking budgets and patch-level mask rankings, thereby bridging this gap.

3 The Proposed DIAMOND

Problem Definition. We consider multivariate time-series forecasting with adaptive patch masking. Let $\mathcal{D} = \{(\boldsymbol{X}^{(n)}, \boldsymbol{Y}^{(n)})\}_{n=1}^{N}$ denote a dataset of N samples, where $\boldsymbol{X}^{(n)} \in \mathbb{R}^{L\times C}$ contains L historical time steps and C variables, and $\boldsymbol{Y}^{(n)} \in \mathbb{R}^{H\times C}$ contains the subsequent H time steps. Each input is divided into $P = L/S$ non-overlapping patches of length S. A masking policy $\pi_{\boldsymbol{\phi}}(\cdot)$ predicts a sample-specific mask ratio $\widehat{\rho}^{(n)}$ and patch scores $\boldsymbol{s}^{(n)}$, which determine a training mask $\boldsymbol{m}^{(n)} \in \{0,1\}^P$; $m_i^{(n)} = 1$ indicates that patch i is masked. A forecaster $f_{\boldsymbol{\theta}}(\cdot)$ predicts $\widehat{\boldsymbol{Y}}^{(n)} = f_{\boldsymbol{\theta}}(\boldsymbol{X}^{(n)}; \boldsymbol{m}^{(n)})$, and $\ell_{\text{pred}}^{(n)}(\boldsymbol{m}) = \|f_{\boldsymbol{\theta}}(\boldsymbol{X}^{(n)}; \boldsymbol{m}) - \boldsymbol{Y}^{(n)}\|_F^2/(HC)$. During training, the controller selects sample-dependent masks with a predefined upper bound on their size, while counterfactual auditing provides patch-level feedback on mask placement.

3.1 Framework Overview

Our study addresses adaptive patch masking for time-series forecasting, aiming to suppress redundant historical information while preserving the temporal patterns required for accurate future prediction. This involves two key challenges: (1) *Sample-Adaptive Masking Amount*. (2) *Task-Aligned*

Masking Supervision. To address these challenges, we propose DIAMOND, which integrates forecasting with adaptive mask control. A shared encoder feeds a forecasting head and a conditional diffusion head. By reconstructing masked patches from the visible context, the diffusion head converts denoising errors into patch-level recoverability evidence, whose aggregation guides sample-specific mask-ratio estimation. A hierarchical controller then employs a scalar head to produce a sample-specific mask ratio from the global representation and aggregated recoverability, and a direction head to estimate patch maskability from local representations and patch-level evidence. The resulting ratio and scores determine the budget and ranking for safety-gated Top-k masking. Periodically, counterfactual auditing restores masked patches individually and measures paired forecasting-loss changes under fixed evaluation conditions. These changes supervise the direction head through pairwise rankings, and update trend-based gates to exclude increasingly harmful patches. The overall architecture of DIAMOND is illustrated in Figure 1, with detailed functionalities of each component elaborated in subsequent sections.

3.2 Diffusion-based Reconstruction for Patch Redundancy Prior

The amount of history that can be masked varies across samples and partly depends on how readily masked patches can be recovered with the help of the remaining history. Therefore, we use the noise-prediction error of a conditional diffusion head as a training-time signal for guiding relative masking amounts. The denoiser observes both the noisy target patch $\boldsymbol{x}_{i,t}$ and the visible context $\boldsymbol{c}_{\mathcal{V}}$.

Denoising Difficulty. Let $\boldsymbol{x}_i \in \mathbb{R}^{S\times C}$ denote the i-th patch. Given a training mask $\boldsymbol{m}$, let $\mathcal{M}(\boldsymbol{m})$ and $\mathcal{V}(\boldsymbol{m})$ denote the masked and visible patch index sets, respectively. The same mask determines the input to the forecaster and the visible context for diffusion reconstruction. For each $i \in \mathcal{M}(\boldsymbol{m})$, we sample a step t from the diffusion-step set $\mathcal{T}$ and noise $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. The noisy patch is:

$$\boldsymbol{x}_{i,t} = \sqrt{\bar{\alpha}_t}\, \boldsymbol{x}_i + \sqrt{1-\bar{\alpha}_t}\, \boldsymbol{\epsilon}_i, \tag{1}$$

where $\bar{\alpha}_t$ is the cumulative signal-retention coefficient. Let $E_{\boldsymbol{\theta}_e}$ denote the shared encoder in $f_{\boldsymbol{\theta}}$, and let $\boldsymbol{c}_{\mathcal{V}}$ denote its representation of the visible patches. Conditioned on this representation, the diffusion head predicts $\widehat{\boldsymbol{\epsilon}}_i = D_{\boldsymbol{\theta}_d}(\boldsymbol{x}_{i,t}, t, \boldsymbol{c}_{\mathcal{V}})$. The patch-level denoising difficulty is:

$$d_i = \frac{1}{SC} \left\| \boldsymbol{\epsilon}_i - \widehat{\boldsymbol{\epsilon}}_i \right\|_F^2 . \tag{2}$$

For one sequence, we average this difficulty over its masked patches:

$$\ell_{\text{diff}}(\boldsymbol{m}) = \frac{1}{\max\{1, |\mathcal{M}(\boldsymbol{m})|\}} \sum_{i\in\mathcal{M}(\boldsymbol{m})} d_i. \tag{3}$$

Let $\mathcal{B}$ denote a minibatch. We define $\mathcal{L}_{\text{diff}} = |\mathcal{B}|^{-1} \sum_{n\in\mathcal{B}} \ell_{\text{diff}}^{(n)}(\boldsymbol{m}^{(n)})$.

Recoverability Evidence. The induced clean-patch estimate is $\widehat{\boldsymbol{x}}_{i,0} = (\boldsymbol{x}_{i,t} - \sqrt{1-\bar{\alpha}_t}\, \widehat{\boldsymbol{\epsilon}}_i)/\sqrt{\bar{\alpha}_t}$. For any fixed non-degenerate step satisfying $0 < \bar{\alpha}_t < 1$, it satisfies:

$$\frac{1}{SC} \left\| \boldsymbol{x}_i - \widehat{\boldsymbol{x}}_{i,0} \right\|_F^2 = \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} d_i. \tag{4}$$

Thus, at a fixed step, lower expected denoising difficulty corresponds to lower conditional reconstruction error. We use d_i as a recoverability proxy and stabilize repeated observations through:

$$e_i^{(u)} = \gamma e_i^{(u-1)} + (1-\gamma)\, \mathrm{sg}\left(d_i^{(u)}\right), \tag{5}$$

where u is the patch-specific observation count and advances only when patch i is masked, $e_i^{(0)}$ is its initial evidence, $\gamma \in [0, 1)$ is the momentum coefficient, and $\mathrm{sg}(\cdot)$ denotes stop-gradient. Evidence for an unmasked patch remains unchanged. Writing u_i for the current observation count of patch i, we collect $\boldsymbol{e}_{\text{rec}} = [e_1^{(u_1)}, \ldots, e_P^{(u_P)}]^\top$ and compute its sample-level mean as $\bar{e}_{\text{rec}} = P^{-1} \sum_{i=1}^{P} e_i^{(u_i)}$. Lower values indicate lower aggregated denoising error under the sampled steps and observed masking contexts.

3.3 Disentangled Masking Predictors under Ranking Guidance

Diffusion evidence provides a prior for how much history to mask, but recoverability alone does not determine which patches are safe to remove for forecasting. Therefore, we use separate heads for the two decisions. A scalar head estimates the sample-level mask ratio under recoverability ranking guidance, while a direction head ranks patches using counterfactual forecasting-loss changes.

Sample-Level Mask Budget. Let $\boldsymbol{z}_i \in \mathbb{R}^d$ be the position-aware pre-mask representation of patch i, and let $\boldsymbol{z}_g \in \mathbb{R}^d$ be the pooled representation of the complete sample. These representations are used to select the mask and the forecaster processes the masked input. The controller $\pi_{\boldsymbol{\phi}}(\cdot)$ consists of a scalar head $g_{\boldsymbol{\phi}_b}(\cdot)$ and a direction head $g_{\boldsymbol{\phi}_s}(\cdot)$, with $\boldsymbol{\phi} = (\boldsymbol{\phi}_b, \boldsymbol{\phi}_s)$. The scalar head combines the global representation with the sample-level diffusion evidence to produce:

$$\eta = g_{\boldsymbol{\phi}_b}\big([\boldsymbol{z}_g \| \bar{e}_{\text{rec}}]\big), \tag{6}$$

where $[\cdot\|\cdot]$ denotes concatenation. We map it to a proportion using the sigmoid function $\sigma(\cdot)$ and limit the proposed masking ratio by $\rho_{\max} \in (0, 1)$:

$$\widehat{\rho} = \rho_{\max}\sigma(\eta), \tag{7}$$

where $\sigma(\cdot)$ is the sigmoid function and $0 < \rho_{\max} < 1$ is the maximum allowed ratio. The upper bound leaves at least one patch visible. To provide relative supervision across samples, let $\mathcal{P}_{\text{rec}}$ contain minibatch pairs (n, n') for which $\bar{e}_{\text{rec}}^{(n)} < \bar{e}_{\text{rec}}^{(n')}$. We define the ratio-ranking loss as:

$$\mathcal{L}_{\text{ratio}} = \frac{1}{\max\{1, |\mathcal{P}_{\text{rec}}|\}} \sum_{(n,n') \in \mathcal{P}_{\text{rec}}} \log\Big(1 + \exp\Big[\eta^{(n')} - \eta^{(n)}\Big]\Big). \tag{8}$$

This encourages samples with lower average denoising difficulty to receive larger predicted ratios.

Patch-Level Selection. For patch selection, the direction head computes $s_i = g_{\boldsymbol{\phi}_s}([\boldsymbol{z}_i \| e_{\text{rec},i}])$, where a larger s_i indicates higher masking priority. Its task-aligned supervision is provided by the counterfactual ranking loss in Section 3.4. Let $q_i \in \{0, 1\}$ denote the latest safety gate for patch i, initialized to one and updated during auditing. We set $\widetilde{s}_i = s_i$ when $q_i = 1$ and $\widetilde{s}_i = -\infty$ otherwise. The number of patches selected is $k = \min\{\lfloor P\widehat{\rho} \rfloor, \sum_{i=1}^{P} q_i\}$. Writing $\widetilde{\boldsymbol{s}} = [\widetilde{s}_1, \ldots, \widetilde{s}_P]^\top$, we construct the binary mask as:

$$m_i = \mathbb{I}[i \in \text{TopK}\,(\widetilde{\boldsymbol{s}}, k)]\,, \tag{9}$$

where $\mathbb{I}[\cdot]$ is the indicator function and $\text{TopK}(\widetilde{\boldsymbol{s}}, k)$ returns the indices of the k highest-scoring eligible patches, or the empty set when $k = 0$. Thus, $\widehat{\rho}$ specifies a proposed budget, $\boldsymbol{s}$ ranks patches within it, and $\boldsymbol{q}$ removes patches excluded by auditing. The realized mask ratio is k/P and can be smaller than $\widehat{\rho}$ because of the safety gates. Given the resulting mask, the forecasting loss is:

$$\mathcal{L}_{\text{pred}} = \frac{1}{|\mathcal{B}|} \sum_{n \in \mathcal{B}} \ell_{\text{pred}}^{(n)}(\boldsymbol{m}^{(n)}), \tag{10}$$

where $\mathcal{B}$ denotes a minibatch. We combine forecasting and diffusion training in the base objective $\mathcal{L}_{\text{base}} = \mathcal{L}_{\text{pred}} + \lambda_{\text{diff}}\mathcal{L}_{\text{diff}}$, where $\lambda_{\text{diff}} \geq 0$ controls the contribution of diffusion reconstruction.

3.4 Counterfactual Variation Auditing with Patch Recovery

Diffusion evidence guides the proposed masking budget, whereas the direction head must learn which patches are safe to mask for forecasting. Therefore, we audit selected patches by comparing predictions before and after restoring one patch.

Patch Restoration. At audit round a, let $\mathcal{A}^{(a)} \subseteq \mathcal{M}(\boldsymbol{m})$ be the set of sampled masked-patch indices. For each $i \in \mathcal{A}^{(a)}$, the mask $\boldsymbol{m}^{(-i)}$ sets m_i to zero and retains all other mask values. With the forecaster and its stochastic state fixed within each comparison, the effect of masking patch i is:

$$\Delta_i^{(a)} = \ell_{\text{pred}}(\boldsymbol{m}) - \ell_{\text{pred}}(\boldsymbol{m}^{(-i)}). \tag{11}$$

A positive $\Delta_i^{(a)}$ means that restoring patch i reduces forecasting loss, so masking it is harmful under the current mask.

Safety Gating. For each audited patch, we maintain a smoothed effect $\bar{\Delta}_i^{(a)} = \beta\bar{\Delta}_i^{(a-1)} + (1-\beta)\Delta_i^{(a)}$, where $\beta \in [0,1)$ is a momentum coefficient and the first observed effect initializes the average. Unaudited values remain unchanged. Let $\mathcal{H}_i^{(a)}$ be the ordered history of observed Δ_i values through round a, and let $h_i^{(a)} = |\mathcal{H}_i^{(a)}|$ be its length. A one-sided Mann–Kendall test on this history produces a statistic $Z_{i,\mathrm{MK}}^{(a)}$, with larger values indicating a stronger increasing trend. Given a minimum history length $h_{\min}$ and a significance level $\alpha_{\mathrm{MK}} \in (0,1)$, we update the safety gate for an audited patch by:

$$q_i^{(a)} = \begin{cases} 1, & h_i^{(a)} < h_{\min}, \\ \mathbb{I}\left[\bar{\Delta}_i^{(a)} \le 0 \ \wedge \ Z_{i,\mathrm{MK}}^{(a)} \le z_{1-\alpha_{\mathrm{MK}}}\right], & h_i^{(a)} \ge h_{\min}, \end{cases} \tag{12}$$

where $z_{1-\alpha_{\mathrm{MK}}}$ is the one-sided critical value. An unaudited gate retains its previous value. Thus, after sufficient observations, a patch is excluded if its smoothed masking effect is harmful or its effect exhibits a significant increasing trend.

Counterfactual Ranking Supervision. The same effects provide ranking supervision for the direction head. Let $\mathcal{B}_{\mathrm{a}}$ be the set of samples in an audit batch, and let $\mathcal{A}_n^{(a)}$ be the audited patch indices of sample n. We form the set of within-sample ordered pairs $\mathcal{Q}^{(a)} = \{(n,i,j) \mid n \in \mathcal{B}_{\mathrm{a}},\, i,j \in \mathcal{A}_n^{(a)},\, \bar{\Delta}_i^{(n,a)} < \bar{\Delta}_j^{(n,a)}\}$. Because patch i has the less harmful masking effect, its score $s_i^{(n)}$ should exceed $s_j^{(n)}$. The direction-ranking loss is:

$$\mathcal{L}_{\mathrm{dir}} = \frac{1}{\max\{1, |\mathcal{Q}^{(a)}|\}} \sum_{(n,i,j)\in\mathcal{Q}^{(a)}} \log\left(1 + \exp\left[s_j^{(n)} - s_i^{(n)}\right]\right). \tag{13}$$

3.5 Summarization

We first warm up the forecaster and diffusion head for E_{w} epochs using random masks at a fixed ratio ρ_{w}. We then alternate ordinary model updates with controller updates every K_{a} epochs. The overall training objective is:

$$\mathcal{L}_{\mathrm{train}} = \mathcal{L}_{\mathrm{base}} + \lambda_{\mathrm{ratio}}\mathcal{L}_{\mathrm{ratio}} + \mathcal{L}_{\mathrm{dir}}. \tag{14}$$

where $\mathcal{L}_{\mathrm{base}}$ trains the shared encoder, forecasting head, and diffusion head, $\mathcal{L}_{\mathrm{ratio}}$ guides the scalar head with cross-sample recoverability rankings, and $\mathcal{L}_{\mathrm{dir}}$ trains the direction head using counterfactual patch rankings. Ordinary updates optimize only $\mathcal{L}_{\mathrm{base}}$, whereas audit rounds freeze the forecasting and diffusion components, update the controller with the ranking losses, and refresh the safety gates. The detailed training procedure of `DIAMOND` is provided in Algorithm 1.

3.6 Theoretical Analysis

We formalize the connection between the diffusion denoising error and patch recoverability.

Definition 3.1. *For a fixed patch and visible context, define the step-t conditional reconstruction risk:*

$$\mathcal{E}_i\left(t \mid \boldsymbol{c}_{\mathcal{V}}\right) = \mathbb{E}_{\boldsymbol{\epsilon}_i}\left[\frac{1}{SC}\left\|\boldsymbol{x}_i - \widehat{\boldsymbol{x}}_{i,0}\right\|_F^2\right], \tag{15}$$

where the expectation is taken over the injected noise. A patch is considered more conditionally recoverable when $\mathcal{E}_i(t \mid \boldsymbol{c}_{\mathcal{V}})$ is smaller.

Theorem 3.1. *For any fixed t such that $0 < \bar{\alpha}_t < 1$,*

$$\frac{1}{SC}\left\|\boldsymbol{x}_i - \widehat{\boldsymbol{x}}_{i,0}\right\|_F^2 = \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} d_i. \tag{16}$$

Consequently,

$$\mathcal{E}_i\left(t \mid \boldsymbol{c}_{\mathcal{V}}\right) = \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}\mathbb{E}_{\boldsymbol{\epsilon}_i}\left[d_i \mid t, \boldsymbol{c}_{\mathcal{V}}\right]. \tag{17}$$

For fixed t, lower expected denoising error indicates higher conditional recoverability.

Table 1: Forecasting results on open benchmarks. The results are averaged over prediction lengths $H \in \{96, 192, 336, 720\}$ (for ILI, $H \in \{24, 36, 48, 60\}$). The input length is fixed at $L = 96$ (for ILI, $L = 36$). Best in **bold**, second-best underlined.

Model	ETTh1		ETTh2		ETTm1		ETTm2		Weather		Electricity		Solar-Energy		Traffic		ILI	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Crossformer	0.449	0.445	0.795	0.599	0.413	0.404	0.379	0.395	0.241	0.268	0.213	0.290	0.403	0.374	0.920	0.541	5.035	1.542
+SL	0.431	0.441	0.659	0.568	0.388	0.413	0.370	0.408	0.213	0.265	0.168	0.264	0.390	0.362	0.514	0.461	3.681	1.285
+DT	0.441	0.439	0.627	0.522	0.394	0.392	0.351	0.378	0.239	0.266	0.211	0.288	0.395	0.370	0.900	0.535	4.321	1.406
Ours	**0.426**	**0.436**	**0.621**	**0.517**	**0.382**	**0.383**	**0.343**	**0.370**	**0.210**	**0.263**	**0.166**	**0.261**	**0.384**	**0.354**	**0.502**	**0.451**	**3.652**	**1.276**
PatchTST	0.459	0.432	0.384	0.403	0.396	0.387	0.280	0.319	0.251	0.270	0.215	0.291	0.287	0.333	0.403	0.286	3.322	1.110
+SL	0.410	0.417	0.374	0.405	0.352	0.380	0.252	0.312	0.225	0.250	0.165	0.258	0.278	0.322	0.428	0.273	1.905	0.895
+DT	0.446	0.429	0.378	0.398	0.393	0.386	0.280	0.319	0.247	0.266	0.210	0.286	0.280	0.328	0.395	0.282	3.126	1.098
Ours	**0.399**	**0.413**	**0.370**	**0.393**	**0.343**	**0.371**	**0.244**	**0.306**	**0.219**	**0.248**	**0.163**	**0.255**	**0.271**	**0.314**	**0.383**	**0.268**	**1.889**	**0.886**
TimesNet	0.534	0.492	0.480	0.459	0.519	0.474	0.348	0.364	0.289	0.305	0.206	0.304	0.406	0.406	0.382	0.429	6.045	1.303
+SL	0.429	0.439	0.368	0.409	0.361	0.384	0.258	0.316	0.239	0.271	0.191	0.294	0.393	0.393	0.363	0.413	2.154	0.931
+DT	0.520	0.483	0.456	0.447	0.514	0.472	0.348	0.364	0.284	0.303	0.203	0.301	0.398	0.400	0.375	0.422	4.872	1.223
Ours	**0.421**	**0.433**	**0.360**	**0.403**	**0.356**	**0.378**	**0.253**	**0.310**	**0.235**	**0.267**	**0.187**	**0.290**	**0.387**	**0.387**	**0.356**	**0.406**	**2.117**	**0.919**
iTransformer	0.447	0.432	0.384	0.400	0.400	0.390	0.284	0.321	0.268	0.280	0.215	0.291	0.317	0.347	0.270	0.413	3.163	1.070
+SL	0.415	0.425	0.382	0.414	0.357	0.380	0.256	0.313	0.229	0.257	0.157	0.249	0.310	0.340	0.343	0.399	1.710	0.857
+DT	0.445	0.431	0.381	0.399	0.399	0.390	0.284	0.321	0.267	0.279	0.214	0.274	0.310	0.340	0.265	0.408	2.994	1.048
Ours	**0.405**	**0.416**	**0.376**	**0.396**	**0.354**	**0.376**	**0.252**	**0.306**	**0.224**	**0.252**	**0.155**	**0.247**	**0.307**	**0.337**	**0.261**	**0.390**	**1.674**	**0.844**
TimeMixer	0.458	0.429	0.386	0.403	0.393	0.385	0.278	0.317	0.283	0.302	0.212	0.282	0.275	0.325	0.263	0.393	3.182	1.149
+SL	0.411	0.421	0.381	0.412	0.351	0.374	0.251	0.309	0.226	0.268	0.160	0.254	0.270	0.320	0.335	0.394	2.026	0.895
+DT	0.455	0.428	0.380	0.399	0.392	0.383	0.278	0.317	0.244	0.264	0.212	0.282	0.265	0.315	0.255	0.385	3.173	1.148
Ours	**0.398**	**0.413**	**0.370**	**0.395**	**0.347**	**0.370**	**0.245**	**0.302**	**0.219**	**0.259**	**0.156**	**0.252**	**0.262**	**0.311**	**0.249**	**0.376**	**1.991**	**0.881**
Dynamic TMoE	0.429	0.430	0.368	0.398	0.376	0.394	0.269	0.318	0.240	0.270	0.170	0.268	0.216	0.280	0.479	0.288	1.981	0.888
+SL	0.425	0.427	0.365	0.395	0.373	0.391	0.267	0.316	0.238	0.268	0.169	0.266	0.212	0.275	0.465	0.280	1.968	0.882
+DT	0.421	0.424	0.362	0.393	0.370	0.389	0.265	0.314	0.237	0.267	0.168	0.265	0.209	0.271	0.450	0.273	1.954	0.877
Ours	**0.413**	**0.416**	**0.355**	**0.388**	**0.362**	**0.384**	**0.261**	**0.309**	**0.232**	**0.262**	**0.165**	**0.262**	**0.205**	**0.267**	**0.444**	**0.269**	**1.945**	**0.872**

See Appendix C.1 for the proof.

Assumption 3.1. *For a fixed sample and patch i, let u index its observed updates within an evidence-update window, i.e., the updates for which $i \in \mathcal{M}(\boldsymbol{m})$, and write*

$$Z_i^{(u)} = \mathrm{sg}(d_i^{(u)}) = \mu_i + \xi_i^{(u)}.$$

We assume that $\mu_i = \mathbb{E}[Z_i^{(u)}]$ is constant within the window, $e_i^{(0)}$ is deterministic, $\mathbb{E}[\xi_i^{(u)}] = 0$, and $0 < \mathrm{Var}(\xi_i^{(u)}) = \sigma_i^2 < \infty$. Across observed updates, the noise is either independent or, for some $\kappa_i \in [0, 1)$, satisfies

$$\left|\mathrm{Cov}\left(\xi_i^{(r)}, \xi_i^{(s)}\right)\right| \le \sigma_i^2 \kappa_i^{|r-s|}. \tag{18}$$

Theorem 3.2. *Under Assumption 3.1, the update in equation 5 satisfies*

$$\mathbb{E}\left[e_i^{(u)}\right] = \mu_i + \gamma^u \left(e_i^{(0)} - \mu_i\right). \tag{19}$$

If the noise variables are independent, then

$$\mathrm{Var}\left(e_i^{(u)}\right) = \sigma_i^2 \frac{1-\gamma}{1+\gamma}\left(1 - \gamma^{2u}\right). \tag{20}$$

Under the weak-dependence condition in equation 18,

$$\mathrm{Var}\left(e_i^{(u)}\right) \le \sigma_i^2 \frac{1-\gamma}{1+\gamma}\frac{1+\gamma\kappa_i}{1-\gamma\kappa_i}. \tag{21}$$

Consequently, the initialization bias decays geometrically, and for $0 < \gamma < 1$ the EMA has lower variance than a single observation.

See Appendix C.2 for the proof.

4 Experiment

4.1 Experimental Settings

Datasets. Following (Wang et al., 2024), we evaluate on long-term and short-term forecasting. For long-term forecasting, we experiment on 9 benchmarks: ETT datasets (including 4 subsets: ETTh1,

Table 2: Forecasting performance on ILI with varying training-data ratios and zero-shot cross-dataset transfer. In zero-shot transfers, h1, h2, m1, and m2 denote ETTh1, ETTh2, ETTm1, and ETTm2, respectively. Best results are in **bold**.

Scenario	Data	Metric	TimesNet				iTransformer				TimeMixer			
			w/o	+SL	+DT	+Ours	w/o	+SL	+DT	+Ours	w/o	+SL	+DT	+Ours
Few-Shot	10%	MAE	1.565	1.098	1.078	**1.069**	1.285	1.010	0.990	**0.981**	1.380	1.055	1.035	**1.026**
		MSE	9.140	3.367	3.267	**3.236**	4.782	2.674	2.574	**2.541**	4.811	3.168	3.068	**3.039**
	25%	MAE	1.356	0.970	0.980	**0.961**	1.114	0.893	0.903	**0.889**	1.196	0.933	0.943	**0.925**
		MSE	6.992	2.641	2.741	**2.607**	3.659	2.098	2.198	**2.077**	3.680	2.485	2.585	**2.460**
	50%	MAE	1.310	0.938	0.918	**0.911**	1.075	0.863	0.843	**0.840**	1.155	0.901	0.881	**0.875**
		MSE	6.863	2.621	2.521	**2.497**	3.590	2.081	1.981	**1.949**	3.612	2.465	2.365	**2.342**
	75%	MAE	1.304	0.929	0.939	**0.920**	1.071	0.855	0.865	**0.849**	1.150	0.893	0.903	**0.886**
		MSE	6.920	2.587	2.687	**2.552**	3.622	2.054	2.154	**2.026**	3.643	2.434	2.534	**2.407**
	100%	MAE	1.303	0.931	1.223	**0.919**	1.070	0.857	1.048	**0.844**	1.149	0.895	1.148	**0.881**
		MSE	6.045	2.154	4.872	**2.117**	3.163	1.710	2.994	**1.674**	3.182	2.026	3.173	**1.991**
	Average	MAE	1.368	0.973	1.028	**0.956**	1.123	0.896	0.930	**0.881**	1.206	0.935	0.982	**0.919**
		MSE	7.192	2.674	3.218	**2.602**	3.763	2.123	2.380	**2.053**	3.786	2.516	2.745	**2.448**
Zero-Shot	h1→h2	MSE	0.469	0.419	0.415	**0.411**	0.421	0.394	0.388	**0.381**	0.424	0.389	0.386	**0.383**
		MAE	0.465	0.439	0.437	**0.434**	0.434	0.420	0.424	**0.413**	0.438	0.416	0.413	**0.410**
	h1→m2	MSE	0.359	0.351	0.354	**0.348**	0.322	0.310	0.312	**0.301**	0.313	0.300	0.305	**0.295**
		MAE	0.397	0.389	0.393	**0.384**	0.371	0.359	0.356	**0.352**	0.360	0.352	0.355	**0.347**
	h2→h1	MSE	0.828	0.595	0.589	**0.584**	0.622	0.501	0.498	**0.487**	0.679	0.560	0.556	**0.552**
		MAE	0.642	0.517	0.514	**0.512**	0.555	0.487	0.490	**0.481**	0.568	0.510	0.509	**0.507**
	m1→h2	MSE	0.483	0.465	0.467	**0.461**	0.447	0.428	0.433	**0.421**	0.441	0.434	0.437	**0.431**
		MAE	0.485	0.469	0.472	**0.462**	0.456	0.444	0.443	**0.440**	0.453	0.445	0.447	**0.438**
	m1→m2	MSE	0.321	0.288	0.285	**0.282**	0.276	0.271	0.267	**0.258**	0.275	0.267	0.261	**0.256**
		MAE	0.363	0.339	0.338	**0.334**	0.334	0.326	0.328	**0.321**	0.326	0.319	0.317	**0.313**
	m2→m1	MSE	0.727	0.459	0.464	**0.454**	0.554	0.443	0.446	**0.432**	0.451	0.428	0.430	**0.424**
		MAE	0.578	0.448	0.450	**0.442**	0.498	0.439	0.437	**0.435**	0.466	0.425	0.429	**0.419**

ETTh2, ETTm1, ETTm2), Weather, Solar-Energy, Electricity, Traffic and ILI (Zhou et al., 2021; Wu et al., 2021; Liu et al., 2022). More details are provided in Appendix.

Baselines. We evaluate our method against DropoutTS (Zhong et al., 2026) and Selective Learning (Fu et al., 2026) on six recent SOTA backbones with diverse designs: PatchTST (Nie et al., 2023), Crossformer (Zhang & Yan, 2023), TimeMixer (Wang et al., 2024), TimesNet (Wu et al., 2023), iTransformer (Liu et al., 2024b), and DynamicTMoE (Zhu et al., 2026). For a fair comparison, both methods share identical training and evaluation protocols.

Implementation Details and Evaluation Metrics. All methods share the standard long-term forecasting protocol (Wu et al., 2023; Zhong et al., 2026). The prediction horizon is $H \in \{96, 192, 336, 720\}$ except for ILI, where $H \in \{24, 36, 48, 60\}$. The look-back window is fixed at L=96, except ILI (L=36), whose series is too short for L=96. We train with MAE using Adam (Kingma & Ba, 2015) (lr=2×10^{-4}, weight decay 5×10^{-4}), batch size 64 by default. We employed the Mean Squared Error (MSE) and Mean Absolute Error (MAE) as the evaluation metrics. All experiments are implemented by PyTorch.

4.2 Empirical Results

Performance on Long-Term Forecasting. We compare Ours with six raw forecasting backbones and existing plug-in methods on nine benchmark datasets. The results are shown in Table 1. **Obs.❶** *Ours improves the raw backbones in every setting.* For example, it reduces MSE on ILI from 6.045 to 2.117 with TimesNet (65.0%) and on Electricity from 0.215 to 0.163 with PatchTST (24.2%). **Obs.❷** *Ours also outperforms the existing plug-in methods.* Compared with the best-performing plug-in in each setting, it reduces MSE from 0.165 to 0.163 on PatchTST with Electricity (1.2%) and from 0.209 to 0.205 on Dynamic TMoE with Solar-Energy (1.9%). **Obs.❸** *The gains persist across backbones and datasets.* Ours achieves the lowest MSE and MAE in all 54 backbone–dataset settings, supporting the general applicability of its adaptive masking strategy.

Performance under Few-Shot and Zero-Shot Settings. Furthermore, we report the few-shot and zero-shot forecasting results in Table 2. ❶ Ours performs consistently across training-data ratios. With only 10% of the ILI training data, it reduces MSE from 9.140 to 3.236 on TimesNet, from 4.782 to 2.541 on iTransformer, and from 4.811 to 3.039 on TimeMixer. Compared with the strongest competing results at this ratio, these correspond to further MSE reductions of 0.9%, 1.3%, and 0.9%, respectively. ❷ In the zero-shot setting, Ours achieves the lowest MSE and MAE across all six transfer directions and three backbones. For instance, on ETTh2→ETTh1, it reduces MSE from the best competing results of 0.589, 0.498, and 0.556 to 0.584, 0.487, and 0.552 on TimesNet,

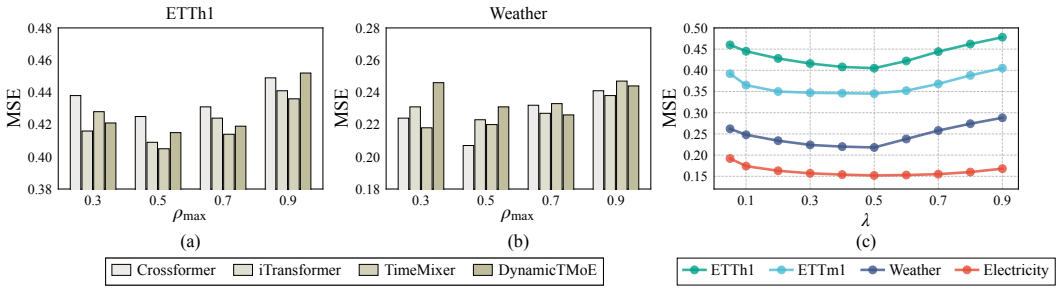


Figure 2: (a) and (b) Comparison of model performance under different maximum mask ratios $\rho_{\max}$ on ETTh1 and Weather. (c) Sensitivity analysis of diffusion loss weights λ on four datasets.

iTransformer, and TimeMixer, respectively. These results demonstrate the effectiveness of Ours with limited training data and under cross-dataset transfer.

Ablation Study. As shown in Table 3, we introduce five variants of Ours on ILI using iTransformer as the backbone. (1) *Random Mask* replaces adaptive masking with random selection; (2) *w/o Diffusion* removes patch recoverability estimation; (3) *w/o Controller* removes sample-level budget estimation and patch-level ranking; (4) *w/o CF Supervision* removes counterfactual supervision; and (5) *w/o Shared Mask* uses different masks for forecasting and diffusion reconstruction. First, removing the controller causes the largest performance degradation, while random masking increases MSE and MAE, demonstrating the value of adaptive masking decisions. Second, removing counterfactual supervision increases both metrics, indicating that recoverability alone cannot determine whether masking a patch harms forecasting. Finally, removing diffusion estimation or mask sharing show that recoverability evidence and a shared mask both contribute to performance.

Table 3: Ablation study.

Method	MSE		MAE	
	Avg	Impv.	Avg	Impv.
Baseline	3.163	–	1.070	–
Random Mask	2.078	↑34.3%	0.941	↑12.1%
w/o Diffusion	1.998	↑36.8%	0.916	↑14.4%
w/o Controller	2.287	↑27.7%	0.986	↑7.9%
w/o CF Supervision	2.118	↑33.0%	0.947	↑11.5%
w/o Shared Mask	2.045	↑35.3%	0.927	↑13.4%
Full (Ours)	1.674	↑47.1%	0.844	↑21.1%

Masking Efficiency. To further evaluate how the upper bound on the controller's masking budget affects forecasting performance, we vary $\rho_{\max}$ across 0.3, 0.5, 0.7, and 0.9 on ETTh1 and Weather using Crossformer, iTransformer, TimeMixer, and DynamicTMoE. Figure 2 (a) and (b) show the MSE. On ETTh1, all backbones achieve their lowest MSE at $\rho_{\max} = 0.5$. On Weather, DynamicTMoE reaches its lowest MSE at 0.7. For every backbone, $\rho_{\max} = 0.9$ yields a higher MSE than its best setting.

Sensitivity Analysis. In this part, we conduct a sensitivity analysis on the diffusion weight λ_{diff}. Figure 2 (c) illustrates its impact on MSE on four datasets: ETTh1, ETTm1, Weather, and Electricity. We vary λ_{diff} in $[0.05, 0.9]$. The results show that all four datasets achieve the best performance at λ_{diff}=0.5 within the evaluated parameter range. A smaller weight yields a weak recoverability signal for the controller, whereas a larger weight lets the diffusion objective dominate $\mathcal{L}_{\text{pred}}$ and harms forecasting. Therefore, we set λ_{diff}=0.5 throughout the main experiments.

5 Conclusion

This paper presents `DIAMOND`, a diffusion-informed patch masking framework for time-series forecasting. `DIAMOND` decomposes mask generation into sample-level masking budget estimation and patch-level safety ranking. It further employs counterfactual restoration to align diffusion-based recoverability evidence with actual forecasting impact. Extensive experiments across multiple benchmarks and forecasting backbones validate its effectiveness and generalizability. In addition to strong empirical performance, `DIAMOND` demonstrates robustness across varying forecasting horizons and backbone architectures, making it broadly applicable to real-world time-series forecasting tasks.

REFERENCES

Jiaxiang Dong, Haixu Wu, Haoran Zhang, Li Zhang, Jianmin Wang, and Mingsheng Long. Simmtm: A simple pre-training framework for masked time-series modeling. *Advances in Neural Information Processing Systems*, 36:29996–30025, 2023.

Jiaxiang Dong, Haixu Wu, Yuxuan Wang, Yunzhong Qiu, Li Zhang, Jianmin Wang, and Mingsheng Long. Timesiam: A pre-training framework for siamese time-series modeling. *arXiv preprint arXiv:2402.02475*, 2024.

Xinyao Fan, Yueying Wu, Chang Xu, Yu-Hao Huang, Weiqing Liu, and Jiang Bian. Mg-tsd: Multi-granularity time series diffusion models with guided learning process. In *International Conference on Learning Representations*, volume 2024, pp. 12916–12934, 2024.

Yisong Fu, Zezhi Shao, Chengqing Yu, Yujie Li, Zhulin An, Qi Wang, Yongjun Xu, and Fei Wang. Selective learning for deep time series forecasting. *Advances in neural information processing systems*, 38:98084–98115, 2026.

Georg Goerg. Forecastable component analysis. In *International conference on machine learning*, pp. 64–72. PMLR, 2013.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015.

Marcel Kollovieh, Abdul Fatir Ansari, Michael Bohlke-Schneider, Jasper Zschiegner, Hao Wang, and Yuyang Bernie Wang. Predict, refine, synthesize: Self-guiding diffusion models for probabilistic time series forecasting. *Advances in Neural Information Processing Systems*, 36:28341–28364, 2023.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.

Yan Li, Xinjiang Lu, Yaqing Wang, and Dejing Dou. Generative time series forecasting with diffusion, denoise, and disentanglement. *Advances in Neural Information Processing Systems*, 35: 23009–23022, 2022.

Renzhao Liang, Sizhe Xu, Chenggang Xie, Jingru Chen, Feiyang Ren, Shu Yang, and Takahiro Yabe. Abstain mask retain core: Time series prediction by adaptive masking loss with representation consistency. *Advances in Neural Information Processing Systems*, 38:99445–99471, 2026.

Jingwei Liu, Ling Yang, Hongyan Li, and Shenda Hong. Retrieval-augmented diffusion models for time series forecasting. *Advances in Neural Information Processing Systems*, 37:2766–2786, 2024a.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia LAI, Lingna Ma, and Qiang Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. In *Advances in Neural Information Processing Systems*, volume 35, pp. 5816–5828. Curran Associates, Inc., 2022.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *International conference on learning representations*, volume 2024, pp. 11116–11140, 2024b.

Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer-xl: Long-context transformers for unified time series forecasting. In *International Conference on Learning Representations*, volume 2025, pp. 83982–84006, 2025.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

Tianyu Qiu, Yi Xie, Hao Niu, Yun Xiong, and Xiaofeng Gao. Enhancing masked time-series modeling via dropping patches. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 20077–20085, 2025.

Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International conference on machine learning*, pp. 8857–8868. PMLR, 2021.

Zineb Senane, Lele Cao, Valentin Leonhard Buchner, Yusuke Tashiro, Lei You, Pawel Andrzej Herman, Mats Nordahl, Ruibo Tu, and Vilhelm Von Ehrenheim. Self-supervised learning of time series representation via diffusion process and imputation-interpolation-forecasting mask. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 2560–2571, 2024.

Lifeng Shen and James Kwok. Non-autoregressive conditional diffusion models for time series prediction. In *International Conference on Machine Learning*, pp. 31016–31029. PMLR, 2023.

Edgardo Solano Carrillo, Sreerag Vadakkemeppully Naveenachandran, and Julia Niebling. Tedm: Time series forecasting with elucidated diffusion models. In *International Conference on Learning Representations*, volume 2026, pp. 6696–6726, 2026.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. In *International conference on learning representations*, volume 2024, pp. 38626–38652, 2024.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

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*, volume 34, pp. 22419–22430. Curran Associates, Inc., 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2023.

Yuxuan Yang, Dalin Zhang, Yuxuan Liang, Hua Lu, Gang Chen, and Huan Li. Not all data are good labels: On the self-supervised labeling for time series forecasting. *Advances in Neural Information Processing Systems*, 38:83566–83601, 2026.

Weiwei Ye, Zhuopeng Xu, and Ning Gui. Non-stationary diffusion for probabilistic time series forecasting. *arXiv preprint arXiv:2505.04278*, 2025.

George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 2114–2124, 2021.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023.

Zhiyuan Zhao, Haoxin Liu, and B Aditya Prakash. Tackling time-series forecasting generalization via mitigating concept drift. In *International Conference on Learning Representations*, volume 2026, pp. 93406–93423, 2026.

Siru Zhong, Yiqiu Liu, Zhiqing Cui, Zezhi Shao, Fei Wang, Qingsong Wen, and Yuxuan Liang. Dropoutts: Sample-adaptive dropout for robust time series forecasting. *arXiv preprint arXiv:2601.21726*, 2026.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

Jiawen Zhu, Shuhan Liu, Di Weng, and Yingcai Wu. Dynamic TMoe: A drift-aware dynamic mixture of experts framework for non-stationary time series forecasting. In *Forty-third International Conference on Machine Learning*, 2026.

A STATEMENT ON THE USE OF LARGE LANGUAGE MODELS

During the preparation of this manuscript, we used the Large Language Model (LLM) to polish the language and correct grammatical errors to improve readability. The LLM was not involved in any core research aspects of the paper, such as research ideation, experimental design, or analysis of results. The authors take full responsibility for the accuracy and integrity of the presented work.

B ALGORITHM

Algorithm 1: Procedure of `DIAMOND`

Input: Training set $\mathcal{D}$; forecaster $f_{\boldsymbol{\theta}}$ with shared encoder; diffusion head $D_{\boldsymbol{\theta}_d}$; controller $\pi_{\boldsymbol{\phi}} = \{g_{\boldsymbol{\phi}_b}, g_{\boldsymbol{\phi}_s}\}$; $\rho_{\max}, \rho_{\mathrm{w}}, E_{\mathrm{w}}, N_{\max}, K_{\mathrm{a}}, \lambda_{\mathrm{diff}}, \lambda_{\mathrm{ratio}}$
Output: Trained forecaster $f_{\boldsymbol{\theta}^*}$
Initialize $\boldsymbol{\theta}, \boldsymbol{\theta}_d, \boldsymbol{\phi}$;
Initialize per-sample recoverability evidence $\boldsymbol{e}_{\mathrm{rec}}^{(n)} \leftarrow \mathbf{0}$ and safety gates $\boldsymbol{q}^{(n)} \leftarrow \mathbf{1}$;
Initialize counterfactual histories for all training samples;
for *epoch* $= 1, \ldots, N_{\max}$ **do**
  for *each minibatch* $\mathcal{B} \subset \mathcal{D}$ **do**
    if *epoch* $\leq E_{\mathrm{w}}$ **then**
      Sample random masks at ratio ρ_{w};
    else
      Compute pre-mask representations $\boldsymbol{z}_i^{(n)}, \boldsymbol{z}_g^{(n)}$ for each $n \in \mathcal{B}$;
      Predict $\widehat{\rho}^{(n)}$ using equation 7 and compute $s_i^{(n)} = g_{\boldsymbol{\phi}_s}([\boldsymbol{z}_i^{(n)} \| e_i^{(n)}])$;
      Generate $\boldsymbol{m}^{(n)}$ from $\widehat{\rho}^{(n)}, \boldsymbol{s}^{(n)}, \boldsymbol{q}^{(n)}$ using equation 9;
    Compute $\mathcal{L}_{\mathrm{pred}}$ and $\mathcal{L}_{\mathrm{diff}}$ using the same masks;
    Update observed $\boldsymbol{e}_{\mathrm{rec}}^{(n)}$ using equation 5;
    Update $\boldsymbol{\theta}, \boldsymbol{\theta}_d$ by minimizing $\mathcal{L}_{\mathrm{base}} = \mathcal{L}_{\mathrm{pred}} + \lambda_{\mathrm{diff}} \mathcal{L}_{\mathrm{diff}}$;
  if *epoch* $= E_{\mathrm{w}}$ ***or*** *(epoch* $> E_{\mathrm{w}}$ ***and*** $(\mathit{epoch} - E_{\mathrm{w}}) \bmod K_{\mathrm{a}} = 0$*)* **then**
    Freeze $\boldsymbol{\theta}, \boldsymbol{\theta}_d$;
    Select an audit minibatch $\mathcal{B}_{\mathrm{a}}$;
    Generate an audit mask $\boldsymbol{m}^{(n)}$ for each $n \in \mathcal{B}_{\mathrm{a}}$ using the current masking rule;
    for *each* $n \in \mathcal{B}_{\mathrm{a}}$ **do**
      Select audited indices $\mathcal{A}_n \subseteq \mathcal{M}(\boldsymbol{m}^{(n)})$;
      Compute $\ell_{\mathrm{pred}}^{(n)}(\boldsymbol{m}^{(n)})$ under a fixed evaluation state;
      for *each* $i \in \mathcal{A}_n$ **do**
        Restore patch i to obtain $\boldsymbol{m}^{(n,-i)}$;
        Compute $\Delta_i^{(n)}$ using equation 11 with the same evaluation state;
        Update $\bar{\Delta}_i^{(n)}$ and its audit history;
      Update audited safety gates $\boldsymbol{q}^{(n)}$ from the smoothed effects and trend tests;
    Construct cross-sample recoverability pairs $\mathcal{P}_{\mathrm{rec}}$ and within-sample counterfactual pairs $\mathcal{Q}$;
    Compute $\mathcal{L}_{\mathrm{ratio}}$ using equation 8 and $\mathcal{L}_{\mathrm{dir}}$ using equation 13;
    Update $\boldsymbol{\phi}$ by minimizing $\mathcal{L}_{\mathrm{ctrl}} = \lambda_{\mathrm{ratio}} \mathcal{L}_{\mathrm{ratio}} + \mathcal{L}_{\mathrm{dir}}$;
    Unfreeze $\boldsymbol{\theta}, \boldsymbol{\theta}_d$;
return $f_{\boldsymbol{\theta}^*}$;

C PROOF OF THEOREM

C.1 PROOF OF THEOREM 3.1

Proof. Substituting the forward process in equation 1 into the induced clean-patch estimate gives

$$\begin{aligned}\widehat{\boldsymbol{x}}_{i,0} &= \frac{\sqrt{\bar{\alpha}_t}\boldsymbol{x}_i + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_i - \sqrt{1-\bar{\alpha}_t} D_{\boldsymbol{\theta}_d}(\boldsymbol{x}_{i,t}, t, \boldsymbol{c}_{\mathcal{V}})}{\sqrt{\bar{\alpha}_t}} \\ &= \boldsymbol{x}_i + \sqrt{\frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}} \left[\boldsymbol{\epsilon}_i - D_{\boldsymbol{\theta}_d}(\boldsymbol{x}_{i,t}, t, \boldsymbol{c}_{\mathcal{V}})\right]. \end{aligned} \tag{22}$$

It follows that

$$\boldsymbol{x}_i - \widehat{\boldsymbol{x}}_{i,0} = \sqrt{\frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}}\left[D_{\boldsymbol{\theta}_d}(\boldsymbol{x}_{i,t}, t, \boldsymbol{c}_{\mathcal{V}}) - \boldsymbol{\epsilon}_i\right]. \tag{23}$$

Taking the squared Frobenius norm, dividing by SC, and using the definition of d_i in equation 2, we obtain

$$\begin{aligned}\frac{1}{SC}\left\|\boldsymbol{x}_i - \widehat{\boldsymbol{x}}_{i,0}\right\|_F^2 &= \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}\frac{1}{SC}\left\|D_{\boldsymbol{\theta}_d}(\boldsymbol{x}_{i,t}, t, \boldsymbol{c}_{\mathcal{V}}) - \boldsymbol{\epsilon}_i\right\|_F^2 \\ &= \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} d_i,\end{aligned} \tag{24}$$

which proves equation 16. Taking the expectation over $\boldsymbol{\epsilon}_i$ on both sides yields

$$\mathcal{E}_i(t \mid \boldsymbol{c}_{\mathcal{V}}) = \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}\mathbb{E}_{\boldsymbol{\epsilon}_i}\left[d_i \mid t, \boldsymbol{c}_{\mathcal{V}}\right], \tag{25}$$

which is equation 17. Finally, because $(1-\bar{\alpha}_t)/\bar{\alpha}_t$ is positive and common to all patches evaluated at the same step, the expected denoising error and the conditional reconstruction risk induce the same ordering. By Definition 3.1, this is also the ordering of conditional recoverability. □

C.2 Proof of Theorem 3.2

Proof. Unrolling the recursion in equation 5 gives

$$e_i^{(u)} = \gamma^u e_i^{(0)} + (1-\gamma)\sum_{r=1}^{u}\gamma^{u-r} Z_i^{(r)}. \tag{26}$$

Taking expectations and using the finite geometric sum yields

$$\begin{aligned}\mathbb{E}\left[e_i^{(u)}\right] &= \gamma^u e_i^{(0)} + (1-\gamma)\mu_i \sum_{r=1}^{u}\gamma^{u-r} \\ &= \gamma^u e_i^{(0)} + \mu_i\left(1-\gamma^u\right) \\ &= \mu_i + \gamma^u\left(e_i^{(0)} - \mu_i\right),\end{aligned} \tag{27}$$

which proves equation 19.

If the observations are independent, $e_i^{(0)}$ is deterministic and the cross-covariance terms vanish. Therefore,

$$\begin{aligned}\mathrm{Var}\left(e_i^{(u)}\right) &= (1-\gamma)^2\sum_{r=1}^{u}\gamma^{2(u-r)}\sigma_i^2 \\ &= \sigma_i^2(1-\gamma)^2\frac{1-\gamma^{2u}}{1-\gamma^2} \\ &= \sigma_i^2\frac{1-\gamma}{1+\gamma}\left(1-\gamma^{2u}\right),\end{aligned} \tag{28}$$

which proves equation 20. Since $0 \le \gamma < 1$, the bias term $\gamma^u(e_i^{(0)} - \mu_i)$ converges to zero. For $0 < \gamma < 1$, the limiting variance is $\sigma_i^2(1-\gamma)/(1+\gamma)$, which is strictly smaller than the variance σ_i^2 of a single observation.

Under the weak-dependence condition, set $a = u - r$ and $b = u - s$. Using equation 18,

$$\begin{aligned}\mathrm{Var}\left(e_i^{(u)}\right) &= (1-\gamma)^2\sum_{r=1}^{u}\sum_{s=1}^{u}\gamma^{2u-r-s}\,\mathrm{Cov}\left(\xi_i^{(r)}, \xi_i^{(s)}\right) \\ &\le \sigma_i^2(1-\gamma)^2\sum_{a=0}^{u-1}\sum_{b=0}^{u-1}\gamma^{a+b}\kappa_i^{|a-b|} \\ &\le \sigma_i^2(1-\gamma)^2\left[\frac{1}{1-\gamma^2} + \frac{2}{1-\gamma^2}\sum_{d=1}^{\infty}(\gamma\kappa_i)^d\right] \\ &= \sigma_i^2\frac{1-\gamma}{1+\gamma}\frac{1+\gamma\kappa_i}{1-\gamma\kappa_i},\end{aligned} \tag{29}$$

which proves equation 21. For $0 < \gamma < 1$ and $0 \leq \kappa_i < 1$, the multiplicative factor is smaller than one because

$$(1+\gamma)(1-\gamma\kappa_i) - (1-\gamma)(1+\gamma\kappa_i) = 2\gamma(1-\kappa_i) > 0. \tag{30}$$

□

D Datasets Details

- **ETT (Electricity Transformer Temperature).** The ETT dataset (Zhou et al., 2021) contains electricity transformer data collected from two counties in China from July 2016 to July 2018. Each time step includes six power load features and one target variable, "oil temperature". The dataset has four subsets: ETTh1 and ETTh2 are recorded hourly, while ETTm1 and ETTm2 are recorded every 15 minutes. Each subset has 7 variables. We follow the standard split: ETTh1/ETTh2 use $(8545, 2881, 2881)$ for training, validation, and testing, while ETTm1/ETTm2 use $(34465, 11521, 11521)$. The prediction horizons are $H \in \{96, 192, 336, 720\}$. The input length is fixed at $L = 96$ for all ETT subsets. The forecastability scores, measured as one minus the entropy of the Fourier decomposition (Goerg, 2013), are approximately 0.38 for ETTh1, 0.45 for ETTh2, 0.46 for ETTm1, and 0.55 for ETTm2, indicating that ETTh1/ETTh2 are relatively more challenging.
- **Weather.** The Weather dataset (Wu et al., 2021) contains 21 meteorological indicators collected every 10 minutes in 2020 from the Weather Station of the Max Planck Biogeochemistry Institute. It has 21 variables. The dataset size is $(36792, 5271, 10540)$ for training, validation, and testing. The prediction horizons are $H \in \{96, 192, 336, 720\}$ with input length $L = 96$. The forecastability is about 0.75, suggesting relatively high predictability.
- **Electricity (ECL).** The Electricity dataset (Wu et al., 2021) records the hourly electricity consumption of 321 clients from 2012 to 2014. It has 321 variables. The standard split is $(18317, 2633, 5261)$ for training, validation, and testing. The prediction horizons are $H \in \{96, 192, 336, 720\}$ with input length $L = 96$. The forecastability is approximately 0.77.
- **Solar-Energy.** The Solar-Energy dataset (Lai et al., 2018) records solar power production from 137 PV plants in 2006, sampled every 10 minutes. It has 137 variables. The dataset size is $(36601, 5161, 10417)$ for training, validation, and testing. The prediction horizons are $H \in \{96, 192, 336, 720\}$ with input length $L = 96$. The forecastability is about 0.33, indicating a challenging benchmark with low predictability.
- **Traffic.** The Traffic dataset (Wu et al., 2021) collects hourly road occupancy rates measured by 862 sensors on San Francisco Bay Area freeways from January 2015 to December 2016. It has 862 variables. The dataset size is $(12185, 1757, 3509)$ for training, validation, and testing. The prediction horizons are $H \in \{96, 192, 336, 720\}$ with input length $L = 96$. The forecastability is about 0.68.
- **ILI.** The ILI dataset (Wu et al., 2023) collects the ratio of influenza-like illness patients versus total patients on a weekly basis, reported by the Centers for Disease Control and Prevention (CDC) of the United States from 2002 to 2021. It has 7 variables. Following common practice, we use a dataset split of approximately $(617, 74, 170)$ for training, validation, and testing. The prediction horizons are $H \in \{24, 36, 48, 60\}$. The input length is set to $L = 36$ for ILI. The forecastability is relatively low, making it a challenging dataset for long-term forecasting.

E Baselines Details

We compare our method with six representative backbone models and two existing plugins. The backbones cover Transformer-based, MLP-based, and general task-oriented designs.

Backbone Baselines

- **PatchTST (Nie et al., 2023).** PatchTST is a channel-independent Transformer for multivariate long-term forecasting. It splits each univariate series into subseries-level patches, which are used as input tokens to a vanilla Transformer encoder. This design preserves local semantic information, reduces the quadratic complexity of attention, and allows the model to attend to longer look-back windows. PatchTST also supports masked self-supervised pretraining and transfer learning. We include it as a strong Transformer-based baseline with patching and channel independence.

- **Crossformer (Zhang & Yan, 2023).** Crossformer explicitly models cross-dimension dependencies for multivariate time series forecasting. It introduces a two-stage attention mechanism, *i.e.*, cross-time attention and cross-dimension attention, together with a hierarchical encoder-decoder architecture. It also adopts a dimension-segment-wise (DSW) embedding to handle multivariate inputs. We include Crossformer as a representative method that explicitly captures inter-variable correlations.
- **TimesNet (Wu et al., 2023).** TimesNet transforms 1D time series into a set of 2D tensors based on multiple discovered periods. It uses TimesBlock to capture intra-period and inter-period variations with a parameter-efficient Inception block, and adaptively aggregates representations from different periods. TimesNet is a general backbone for multiple time-series tasks, including forecasting, imputation, classification, and anomaly detection. We use it as a general-purpose CNN/Inception-based baseline.
- **iTransformer (Liu et al., 2024b).** iTransformer inverts the dimensions of the vanilla Transformer for multivariate forecasting. Instead of embedding time steps as tokens, it embeds the whole time series of each variate as a variate token. Self-attention is then applied on variate tokens to capture multivariate correlations, while the feed-forward network learns series representations. Layer normalization is applied on each variate token to reduce discrepancies among variables. It is particularly effective on high-dimensional time series and serves as a strong inverted Transformer baseline.
- **TimeMixer (Wang et al., 2024).** TimeMixer is a fully MLP-based multiscale forecasting model. It first downsamples the input into multiple scales, then uses Past-Decomposable-Mixing (PDM) to decompose and mix seasonal and trend components across scales. Seasonal components are mixed bottom-up, while trend components are mixed top-down. Future-Multipredictor-Mixing (FMM) ensembles multiple predictors from different scales for the final prediction. We include TimeMixer as a strong MLP-based multiscale baseline.
- **DynamicTMoE (Zhu et al., 2026).** DynamicTMoE is a recent dynamic mixture-of-experts model for time series forecasting. It is designed to adaptively allocate model capacity across different temporal patterns. As the full paper is not included in our reference set, we follow the original implementation and citation in our experiments. It serves as a strong dynamic-capacity baseline.

Existing Plug-in methods

- **DropoutTS (Zhong et al., 2026).** DropoutTS is a training-time dropout/masking strategy for time series forecasting. It improves robustness by dropping or masking parts of the input during training while keeping the full history at inference. We insert it at the same position within each backbone as our method and use identical training and evaluation protocols.
- **Selective Learning (Fu et al., 2026).** Selective Learning is a training strategy that screens a subset of timesteps for loss computation to mitigate overfitting in deep time series forecasting. It employs a dual-mask mechanism with an uncertainty mask and an anomaly mask, guiding the model to focus on generalizable timesteps while disregarding non-generalizable ones. We insert it at the same position within each backbone as our method and use identical training and evaluation protocols.

F Hyperparameter Settings.

Prediction-loss conventions. All loss comparisons use predictions and targets after per-channel z-score normalization with statistics estimated from the training split. For sample n and mask $\boldsymbol{m}$, let $r_{h,c}^{(n)}(\boldsymbol{m})$ be the prediction error at horizon step h and channel c. We distinguish the loss used for forecasting updates from the loss used for auditing:

$$\ell_{\mathrm{opt}}^{(n)}(\boldsymbol{m}) = \frac{1}{HC}\sum_{h=1}^{H}\sum_{c=1}^{C}\left|r_{h,c}^{(n)}(\boldsymbol{m})\right|, \qquad \ell_{\mathrm{audit}}^{(n)}(\boldsymbol{m}) = \frac{1}{HC}\sum_{h=1}^{H}\sum_{c=1}^{C}\left(r_{h,c}^{(n)}(\boldsymbol{m})\right)^2. \tag{31}$$

The MSE quantity ℓ_{pred} in the problem definition equals ℓ_{audit}. In ordinary model updates, however, the implemented forecasting term uses the MAE surrogate $L_{\mathrm{pred}}^{\mathrm{opt}} = |\mathcal{B}|^{-1}\sum_{n\in\mathcal{B}}\ell_{\mathrm{opt}}^{(n)}(\boldsymbol{m}^{(n)})$; thus the implemented base objective is $L_{\mathrm{base}}^{\mathrm{impl}} = L_{\mathrm{pred}}^{\mathrm{opt}} + \lambda_{\mathrm{diff}}L_{\mathrm{diff}}$.

Counterfactual auditing uses MSE throughout. Specifically, $\Delta_i^{(a)} = \ell_{\mathrm{audit}}(\boldsymbol{m}) - \ell_{\mathrm{audit}}(\boldsymbol{m}^{(-i)})$; its moving average and Mann–Kendall trend determine the safety gate. The tolerance $\delta = 0.02$

Table 4: Shared DiffusionMask hyperparameters used in all main experiments. Symbols match Section 3; values are the production plugin (`light_rho2`).

Component	Hyperparameter	Value
Patching	Patch length S / stride	16 / 16
	#patches P (L=96)	6
Mask budget	Min / max ratio $\rho_{\min}$, $\rho_{\max}$	0.05, 0.35
	Warm-up epochs E_{w}	5
	Warm-up ratio ρ_{w}	0.08
	Loss tolerance δ	0.02
	Candidate set $\mathcal{R}$	$\{0, 0.05, 0.10, \ldots, 0.35\}$
Diffusion	#steps T	4
	Schedule $\bar{\alpha}_t$	t/T
	Evidence momentum γ	0.9
	Weight λ_{diff}	0.5
Controller	Audit interval K_{a}	4
	Audit probes $\lvert\mathcal{A}\rvert$	4
	Budget loss weight λ_ρ	2.0
	Controller loss weight λ_{ctrl}	0.03
Optimization	Epochs / seed	20 / 42
	Optimizer	Adam
	Learning rate / weight decay	2×10^{-4} / 5×10^{-4}
	Train loss / early stopping	MAE / patience 15

likewise denotes an absolute increase in HC-averaged MSE on the normalized target scale: $\ell_{\mathrm{audit}}(\boldsymbol{m}) - \ell_{\mathrm{audit}}(\mathbf{0}) \le \delta$. The safety gate tests the patch-level effects as specified in Eq. (12); it does not compare them directly with δ. Evaluation reports both MSE and MAE.

G More Experiments

G.1 Full results of Long-term forecasting

Table 5 details the full long-term forecasting performance across nine real-world datasets characterized by diverse periodicities and data qualities. A key finding is +DM's ability to revitalize legacy backbones: on complex datasets like Electricity, it enables iTransformer (Avg MSE $0.215 \rightarrow 0.151$) to outperform the vanilla TimesNet (0.206) and DynamicTMoE (0.170), while PatchTST, TimeMixer, and Crossformer are also improved to 0.159, 0.154, and 0.162, respectively; iTransformer attains 29.8% MSE and 16.5% MAE reduction. We also observe significant efficacy on the ILI dataset, which represents a challenging "small-sample, high-noise" regime. Here, +DM provides critical regularization, boosting TimesNet by 64.5% MSE and 29.0% MAE, and Crossformer by 27.0% and 17.1%, corroborating our claim that spectral sparsity acts as a strong inductive bias aiding generalization when data is scarce. Whether on highly periodic domains (Weather), where TimeMixer improves by 22.3%/14.6% and TimesNet by 19.4%/13.1%, or irregular flows (Traffic/ILI), where Crossformer gains 44.8%/15.9%, the consistent positive improvements indicate that the spectral scorer correctly identifies domain-specific "information-rich" frequencies without manual tuning.

G.2 Full Results of Long-Term Forecasting

Table 5 reports the long-term forecasting results across nine datasets and six backbones. Averaged over prediction lengths, `DIAMOND` reduces both MSE and MAE relative to the corresponding raw backbone in all 54 backbone–dataset settings. On Electricity, it reduces iTransformer's MSE from 0.215 to 0.155 (27.9%) and MAE from 0.291 to 0.247 (15.1%); PatchTST, TimeMixer, and Crossformer achieve average MSEs of 0.163, 0.156, and 0.166, respectively. On ILI, TimesNet obtains MSE and MAE reductions of 65.0% and 29.5%, while Crossformer obtains reductions of 27.5% and

17.3%. The gains also extend to Weather, where TimeMixer improves MSE/MAE by 22.6%/14.2% and TimesNet by 18.7%/12.5%, and to Traffic, where Crossformer improves by 45.4%/16.6%. These results show consistent improvements across datasets and forecasting architectures.

G.3 More experiments on Short-term forecasting

We further evaluate **DIAMOND** on four PEMS datasets with an input length of 96 and a prediction length of 12. As shown in Table 6, applying **DIAMOND** to each of the six forecasting backbones reduces MAE, MAPE, and RMSE relative to its original counterpart on every dataset. Within each backbone group, **DIAMOND** achieves the best result in 69 of the 72 dataset–metric comparisons. For example, on PEMS08, it reduces the MAE and RMSE of TimesNet from 19.01 and 30.65 to 16.73 and 26.97, respectively. These results show that the proposed masking strategy also improves short-term forecasting across different model architectures.

G.4 Sensitivity Analysis

We analyze the sensitivity of **DIAMOND** to two key hyperparameters: the number of patches P and the maximum mask budget $\rho_{\max}$. Experiments are conducted on ETTh1 with $L = 96$, using four representative backbones: iTransformer, TimeMixer, Crossformer, and DynamicTMoE. We vary $P \in \{4, 8, 16, 32\}$ and $\rho_{\max} \in \{0.3, 0.5, 0.7, 0.9\}$, and report the average MSE over $H \in \{96, 192, 336, 720\}$. We observe that $P = 8$ and $\rho_{\max} = 0.5$ consistently yield the best performance across all backbones and horizons. A smaller patch number ($P = 4$) suffers from coarse discretization, where the floor operation $\lfloor P\hat{\rho} \rfloor$ amplifies the quantization error and limits the effective mask budget. A larger patch number ($P = 32$) leads to overly short patches ($S = 3$), which hinders both the diffusion-based recoverability estimation and the counterfactual auditing, resulting in degraded forecasting accuracy. For the mask budget, $\rho_{\max} = 0.5$ strikes the best trade-off between suppressing redundant history and preserving informative temporal patterns. Reducing $\rho_{\max}$ to 0.3 provides insufficient regularization, whereas increasing it to 0.7 or 0.9 introduces harmful masking and consistently degrades performance. These trends are consistent across all backbones, confirming the robustness of **DIAMOND** around the recommended operating point $(P^* = 8, \rho^*_{\max} = 0.5)$.

G.5 Efficiency Analysis

Table 7 presents an efficiency comparison across six forecasting backbones. At inference, **DIAMOND** uses the full historical input and runs only the trained forecaster. Its reported latency therefore matches that of the corresponding vanilla backbone at the precision shown in the table. The accuracy gains come with higher training costs. In the results, **DIAMOND** takes 1.83–2.49 times as long to train as the vanilla backbones and uses more peak GPU memory. Periodic auditing accounts for approximately 3.3–4.1% of total training time; most additional computation comes from pre-mask feature extraction and diffusion reconstruction during regular training updates. Meanwhile, **DIAMOND** achieves the lowest MSE across all six backbones. It reduces MSE by 6.4–8.4% relative to vanilla training and by 2.7–5.7% relative to DropoutTS, while outperforming fixed-ratio random masking in every case. These results suggest that sample-adaptive masking budgets and patch selection may improve forecasting accuracy, with the additional cost concentrated in training and inference latency remaining comparable to the vanilla backbones.

G.6 Case Study

Solar stations share a common irradiance factor, so both look-back and future correlations are nearly uniform in daytime windows (Case 1). When the horizon crosses dusk (Case 2) or the look-back window falls at night while the horizon returns to day (Case 3), the future correlation structure changes markedly. A fixed mask rate treats these regimes equally, even though the set of stations that can be dropped without harm is regime-dependent. On the same look-back windows, encoder attention concentrates on a few variate hubs (vertical bands) instead of reproducing the dense Pearson maps (Fig. 4). We report this as evidence that the backbone performs variate selection under shifting correlation structure, *not* as a claim that shallow layers align with the past and deep layers with the future.

Table 5: Full forecasting results on open benchmarks. Each dataset reports MSE/MAE for prediction lengths $H \in \{96, 192, 336, 720\}$ (for ILI, $H \in \{24, 36, 48, 60\}$). +DM denotes our method. The gray rows show the relative improvement (Δ) of +DM over raw backbones (higher ↑ is better).

Method	H	ETTh1				ETTh2				ETTm1				ETTm2				Weather				Electricity				Solar-Energy				Traffic				ILI			
		raw		+DM		raw		+DM		raw		+DM		raw		+DM		raw		+DM		raw		+DM		raw		+DM		raw		+DM		raw		+DM	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Crossformer	96	0.394	0.404	0.411	0.423	0.339	0.395	0.601	0.502	0.345	0.362	0.370	0.377	0.181	0.264	0.333	0.363	0.166	0.204	0.203	0.254	0.192	0.267	0.159	0.253	0.395	0.370	0.375	0.348	0.900	0.530	0.489	0.441	4.736	1.480	3.528	1.258
	192	0.436	0.431	0.421	0.433	0.438	0.440	0.611	0.512	0.378	0.381	0.380	0.382	0.237	0.306	0.343	0.368	0.208	0.244	0.208	0.259	0.198	0.275	0.164	0.258	0.400	0.372	0.380	0.353	0.915	0.538	0.499	0.448	5.153	1.561	3.627	1.267
	336	0.471	0.453	0.431	0.443	0.491	0.488	0.626	0.522	0.430	0.411	0.385	0.387	0.313	0.367	0.348	0.373	0.260	0.286	0.213	0.267	0.214	0.295	0.169	0.264	0.405	0.375	0.385	0.356	0.925	0.545	0.506	0.454	5.244	1.576	3.697	1.282
	720	0.495	0.492	0.441	0.445	1.912	1.073	0.646	0.532	0.499	0.462	0.393	0.386	0.785	0.643	0.348	0.376	0.330	0.338	0.216	0.272	0.248	0.323	0.172	0.269	0.412	0.379	0.396	0.359	0.940	0.551	0.514	0.461	5.007	1.551	3.756	1.297
	Avg	**0.449**	**0.445**	**0.426**	**0.436**	**0.795**	**0.599**	**0.621**	**0.517**	**0.413**	**0.404**	**0.382**	**0.383**	**0.379**	**0.395**	**0.343**	**0.370**	**0.241**	**0.268**	**0.210**	**0.263**	**0.213**	**0.290**	**0.166**	**0.261**	**0.403**	**0.374**	**0.384**	**0.354**	**0.920**	**0.541**	**0.502**	**0.451**	**5.035**	**1.542**	**3.652**	**1.276**
	Imp.	5.1	2.0	5.1	2.0	21.9	13.7	21.9	13.7	7.5	5.2	7.5	5.2	9.5	6.3	9.5	6.3	12.9	1.9	12.9	1.9	22.1	10.0	22.1	10.0	4.7	5.3	4.7	5.3	45.4	16.6	45.4	16.6	27.5	17.3	27.5	17.3
PatchTST	96	0.394	0.392	0.385	0.401	0.293	0.338	0.356	0.381	0.327	0.348	0.332	0.362	0.175	0.252	0.238	0.300	0.173	0.208	0.210	0.239	0.194	0.268	0.154	0.248	0.275	0.325	0.259	0.307	0.395	0.280	0.376	0.263	3.633	1.079	1.840	0.870
	192	0.447	0.423	0.395	0.412	0.376	0.391	0.367	0.391	0.374	0.370	0.342	0.372	0.240	0.296	0.243	0.305	0.215	0.248	0.216	0.246	0.199	0.276	0.159	0.253	0.282	0.330	0.266	0.312	0.400	0.284	0.380	0.266	4.019	1.192	1.870	0.880
	336	0.490	0.444	0.405	0.417	0.425	0.431	0.377	0.396	0.406	0.395	0.349	0.377	0.301	0.335	0.248	0.310	0.269	0.288	0.222	0.252	0.214	0.292	0.164	0.258	0.288	0.335	0.272	0.316	0.405	0.288	0.385	0.270	2.939	1.099	1.900	0.890
	720	0.505	0.469	0.411	0.422	0.442	0.452	0.380	0.404	0.477	0.435	0.349	0.373	0.404	0.393	0.247	0.309	0.347	0.336	0.228	0.255	0.253	0.328	0.175	0.261	0.303	0.342	0.287	0.321	0.412	0.292	0.391	0.273	2.697	1.070	1.946	0.904
	Avg	**0.459**	**0.432**	**0.399**	**0.413**	**0.384**	**0.403**	**0.370**	**0.393**	**0.396**	**0.387**	**0.343**	**0.371**	**0.280**	**0.319**	**0.244**	**0.306**	**0.251**	**0.270**	**0.219**	**0.248**	**0.215**	**0.291**	**0.163**	**0.255**	**0.287**	**0.333**	**0.271**	**0.314**	**0.403**	**0.286**	**0.383**	**0.268**	**3.322**	**1.110**	**1.889**	**0.886**
	Imp.	13.1	4.4	13.1	4.4	3.6	2.5	3.6	2.5	13.4	4.1	13.4	4.1	12.9	4.1	12.9	4.1	12.7	8.1	12.7	8.1	24.2	12.4	24.2	12.4	5.6	5.7	5.6	5.7	5.0	6.3	5.0	6.3	43.1	20.2	43.1	20.2
TimesNet	96	0.488	0.475	0.408	0.425	0.448	0.428	0.348	0.395	0.455	0.439	0.346	0.370	0.207	0.282	0.246	0.302	0.181	0.227	0.227	0.257	0.174	0.279	0.182	0.284	0.395	0.398	0.378	0.380	0.370	0.420	0.347	0.397	9.241	1.389	2.070	0.904
	192	0.513	0.478	0.418	0.430	0.520	0.470	0.358	0.400	0.524	0.472	0.353	0.375	0.289	0.332	0.251	0.308	0.246	0.283	0.232	0.262	0.188	0.288	0.185	0.288	0.400	0.402	0.382	0.384	0.378	0.425	0.353	0.402	7.371	1.438	2.100	0.914
	336	0.581	0.505	0.426	0.435	0.497	0.475	0.363	0.405	0.543	0.489	0.361	0.382	0.362	0.378	0.256	0.312	0.311	0.324	0.238	0.267	0.210	0.307	0.188	0.292	0.408	0.408	0.388	0.388	0.385	0.432	0.359	0.409	4.175	1.237	2.129	0.924
	720	0.554	0.510	0.432	0.442	0.455	0.463	0.371	0.412	0.554	0.496	0.364	0.385	0.534	0.464	0.259	0.318	0.418	0.386	0.243	0.282	0.252	0.342	0.193	0.296	0.421	0.416	0.400	0.396	0.395	0.439	0.365	0.416	3.393	1.148	2.169	0.934
	Avg	**0.534**	**0.492**	**0.421**	**0.433**	**0.480**	**0.459**	**0.360**	**0.403**	**0.519**	**0.474**	**0.356**	**0.378**	**0.348**	**0.364**	**0.253**	**0.310**	**0.289**	**0.305**	**0.235**	**0.267**	**0.206**	**0.304**	**0.187**	**0.290**	**0.406**	**0.406**	**0.387**	**0.387**	**0.382**	**0.429**	**0.356**	**0.406**	**6.045**	**1.303**	**2.117**	**0.919**
	Imp.	21.2	12.0	21.2	12.0	25.0	12.2	25.0	12.2	31.4	20.3	31.4	20.3	27.3	14.8	27.3	14.8	18.7	12.5	18.7	12.5	9.2	4.6	9.2	4.6	4.7	4.7	4.7	4.7	6.8	5.4	6.8	5.4	65.0	29.5	65.0	29.5
Transformer	96	0.384	0.391	0.391	0.407	0.298	0.340	0.366	0.388	0.330	0.351	0.343	0.370	0.180	0.257	0.244	0.299	0.185	0.217	0.215	0.243	0.193	0.267	0.149	0.239	0.308	0.340	0.299	0.331	0.260	0.405	0.252	0.382	3.507	1.071	1.621	0.828
	192	0.438	0.422	0.401	0.412	0.374	0.388	0.373	0.393	0.379	0.376	0.351	0.374	0.245	0.298	0.250	0.304	0.235	0.258	0.221	0.249	0.198	0.275	0.154	0.244	0.315	0.345	0.305	0.335	0.265	0.410	0.257	0.387	3.974	1.152	1.650	0.838
	336	0.487	0.446	0.411	0.419	0.433	0.430	0.379	0.398	0.416	0.399	0.358	0.380	0.305	0.336	0.255	0.309	0.291	0.299	0.227	0.255	0.214	0.292	0.159	0.250	0.322	0.350	0.311	0.341	0.272	0.416	0.264	0.393	2.513	1.005	1.690	0.848
	720	0.479	0.469	0.417	0.426	0.431	0.442	0.386	0.405	0.475	0.434	0.364	0.380	0.406	0.393	0.259	0.312	0.361	0.346	0.233	0.261	0.255	0.330	0.158	0.255	0.323	0.353	0.313	0.341	0.283	0.421	0.271	0.398	2.658	1.052	1.735	0.862
	Avg	**0.447**	**0.432**	**0.405**	**0.416**	**0.384**	**0.400**	**0.376**	**0.396**	**0.400**	**0.390**	**0.354**	**0.376**	**0.284**	**0.321**	**0.252**	**0.306**	**0.268**	**0.280**	**0.224**	**0.252**	**0.215**	**0.291**	**0.155**	**0.247**	**0.317**	**0.347**	**0.307**	**0.337**	**0.270**	**0.413**	**0.261**	**0.390**	**3.163**	**1.070**	**1.674**	**0.844**
	Imp.	9.4	3.7	9.4	3.7	2.1	1.0	2.1	1.0	11.5	3.6	11.5	3.6	11.3	4.7	11.3	4.7	16.4	10.0	16.4	10.0	27.9	15.1	27.9	15.1	3.2	2.9	3.2	2.9	3.3	5.6	3.3	5.6	47.1	21.1	47.1	21.1
TimeMixer	96	0.401	0.395	0.383	0.403	0.297	0.338	0.360	0.387	0.318	0.342	0.337	0.362	0.173	0.250	0.238	0.295	0.180	0.226	0.214	0.252	0.195	0.263	0.152	0.245	0.268	0.318	0.255	0.304	0.255	0.385	0.242	0.369	3.124	1.136	1.937	0.867
	192	0.443	0.420	0.393	0.410	0.375	0.390	0.367	0.392	0.373	0.372	0.344	0.368	0.239	0.295	0.243	0.300	0.246	0.282	0.219	0.258	0.193	0.267	0.156	0.251	0.274	0.324	0.261	0.310	0.262	0.392	0.248	0.375	3.538	1.214	1.976	0.877
	336	0.492	0.441	0.401	0.416	0.435	0.434	0.373	0.397	0.407	0.394	0.352	0.374	0.300	0.333	0.248	0.305	0.309	0.323	0.222	0.262	0.209	0.283	0.159	0.255	0.279	0.328	0.266	0.314	0.268	0.398	0.254	0.381	3.055	1.130	2.016	0.887
	720	0.496	0.460	0.415	0.423	0.437	0.450	0.380	0.404	0.474	0.432	0.355	0.376	0.400	0.390	0.251	0.308	0.397	0.377	0.221	0.264	0.251	0.315	0.157	0.257	0.279	0.330	0.266	0.316	0.267	0.397	0.252	0.379	3.011	1.116	2.035	0.893
	Avg	**0.458**	**0.429**	**0.398**	**0.413**	**0.386**	**0.403**	**0.370**	**0.395**	**0.393**	**0.385**	**0.347**	**0.370**	**0.278**	**0.317**	**0.245**	**0.302**	**0.283**	**0.302**	**0.219**	**0.259**	**0.212**	**0.282**	**0.156**	**0.252**	**0.275**	**0.325**	**0.262**	**0.311**	**0.263**	**0.393**	**0.249**	**0.376**	**3.182**	**1.149**	**1.991**	**0.881**
	Imp.	13.1	3.7	13.1	3.7	4.1	2.0	4.1	2.0	11.7	3.9	11.7	3.9	11.9	4.7	11.9	4.7	22.6	14.2	22.6	14.2	26.4	10.6	26.4	10.6	4.7	4.3	4.7	4.3	5.3	4.3	5.3	4.3	37.4	23.3	37.4	23.3
ynamicTMoE	96	0.420	0.422	0.403	0.408	0.360	0.390	0.347	0.381	0.368	0.386	0.354	0.376	0.262	0.312	0.254	0.303	0.235	0.265	0.227	0.257	0.165	0.262	0.160	0.257	0.210	0.275	0.200	0.262	0.470	0.282	0.436	0.264	1.960	0.880	1.922	0.864
	192	0.428	0.428	0.411	0.414	0.365	0.395	0.352	0.385	0.374	0.392	0.360	0.382	0.267	0.316	0.259	0.307	0.240	0.270	0.232	0.262	0.170	0.268	0.165	0.262	0.215	0.280	0.204	0.267	0.475	0.286	0.441	0.268	1.975	0.885	1.937	0.869
	336	0.432	0.432	0.416	0.418	0.370	0.400	0.357	0.390	0.379	0.396	0.365	0.387	0.272	0.320	0.264	0.311	0.243	0.272	0.235	0.264	0.172	0.270	0.167	0.264	0.218	0.282	0.207	0.269	0.482	0.290	0.447	0.272	1.985	0.892	1.949	0.876
	720	0.436	0.438	0.422	0.424	0.377	0.407	0.364	0.396	0.383	0.402	0.369	0.391	0.275	0.324	0.267	0.315	0.242	0.273	0.234	0.265	0.173	0.272	0.168	0.265	0.221	0.283	0.209	0.270	0.489	0.294	0.452	0.272	2.004	0.895	1.972	0.879
	Avg	**0.429**	**0.430**	**0.413**	**0.416**	**0.368**	**0.398**	**0.355**	**0.388**	**0.376**	**0.394**	**0.362**	**0.384**	**0.269**	**0.318**	**0.261**	**0.309**	**0.240**	**0.270**	**0.232**	**0.262**	**0.170**	**0.268**	**0.165**	**0.262**	**0.216**	**0.280**	**0.205**	**0.267**	**0.479**	**0.288**	**0.444**	**0.269**	**1.981**	**0.888**	**1.945**	**0.872**
	Imp.	3.7	3.3	3.7	3.3	3.5	2.5	3.5	2.5	3.7	2.5	3.7	2.5	3.0	2.8	3.0	2.8	3.3	3.0	3.3	3.0	2.9	2.2	2.9	2.2	5.1	4.6	5.1	4.6	7.3	6.6	7.3	6.6	1.8	1.8	1.8	1.8

Table 6: Short-term forecasting results on PEMS. All input lengths are 96 and prediction lengths are 12. A lower MAE, MAPE, or RMSE indicates a better prediction. Best in **bold**, second-best underlined.

Method	PEMS03			PEMS04			PEMS07			PEMS08		
	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE
Crossformer	15.64	15.74	25.56	20.38	12.84	32.41	22.54	9.38	35.49	17.56	10.92	27.21
+SL	15.50	15.60	25.33	20.20	12.72	32.12	22.24	9.25	35.01	17.40	10.82	26.97
+DT	15.43	15.53	25.22	20.11	12.67	**31.54**	22.34	9.30	35.17	17.32	10.77	26.84
Ours	**15.22**	**15.32**	**24.87**	**19.83**	**12.49**	31.97	**21.93**	**9.13**	**34.53**	**17.09**	**10.63**	**26.48**
PatchTST	18.95	17.29	30.15	24.86	16.65	40.46	27.87	12.69	42.56	20.35	13.15	31.04
+SL	18.29	16.69	29.10	24.00	16.07	39.06	26.90	12.25	41.08	19.64	12.69	29.96
+DT	18.82	17.17	29.94	24.69	16.53	40.18	27.68	12.60	42.26	20.21	13.06	30.82
Ours	**18.03**	**16.45**	**28.68**	**23.65**	**15.84**	**38.49**	**26.52**	**12.07**	**40.49**	**19.36**	**12.51**	**29.53**
TimesNet	16.41	15.17	26.72	21.63	13.15	34.90	25.12	10.60	40.71	19.01	11.83	30.65
+SL	14.64	13.54	23.84	19.30	11.73	31.14	22.41	9.46	36.32	16.96	10.56	27.35
+DT	16.11	14.89	26.23	21.23	12.91	34.26	24.66	10.41	39.97	18.66	11.61	30.09
Ours	**14.44**	**13.35**	**23.52**	**19.04**	**11.57**	**30.71**	**22.11**	**9.33**	**35.83**	**16.73**	**10.41**	**26.97**
iTransformer	15.28	15.12	24.42	19.68	12.70	31.58	21.18	8.88	34.35	15.72	9.88	25.05
+SL	15.03	**14.67**	24.02	19.36	12.49	31.07	20.84	8.74	33.79	15.47	9.72	24.64
+DT	15.24	15.09	24.36	19.63	12.67	31.51	21.13	8.86	34.27	15.68	9.86	24.99
Ours	**14.82**	14.88	**23.69**	**19.09**	**12.32**	**30.63**	**20.54**	**8.61**	**33.32**	**15.25**	**9.58**	**24.30**
TimeMixer	14.63	14.54	23.28	19.21	12.53	30.92	20.57	8.62	33.59	15.22	9.67	24.26
+SL	14.36	14.27	22.85	18.85	12.30	30.34	20.19	8.46	32.96	**14.72**	9.49	23.81
+DT	14.60	14.51	23.23	19.17	12.50	30.85	20.52	8.60	33.51	15.18	9.65	24.20
Ours	**14.15**	**14.07**	**22.52**	**18.58**	**12.12**	**29.91**	**19.90**	**8.34**	**32.49**	14.94	**9.35**	**23.47**
Dynamic TMoE	15.52	15.35	24.78	20.05	12.95	32.15	21.65	9.15	34.92	16.18	10.15	25.88
+SL	15.41	15.24	24.61	19.91	12.86	31.93	21.50	9.09	34.68	15.95	10.01	25.52
+DT	15.30	15.14	24.43	19.77	12.77	31.70	21.35	9.02	34.43	16.07	10.08	25.70
Ours	**15.09**	**14.92**	**24.09**	**19.49**	**12.59**	**31.25**	**21.05**	**8.89**	**33.95**	**15.73**	**9.87**	**25.16**

G.7 Training Dynamics and Generalization.

To examine how the proposed plugin affects training dynamics and generalization, we train the same backbone with DropoutTS and with Ours, and plot the MSE on the training and testing splits across all epochs (Figure 5). A consistent trend appears on ETTh1, ETTm2, Weather, and Solar. On the training set, DropoutTS can reach a comparable or occasionally lower error, because timestep-level dropout still lets the model fit the observed window aggressively. On the test set, however, models trained with DropoutTS remain at a higher MSE in every later epoch, whereas Ours decreases more rapidly and settles at a lower plateau, matching the TimeMixer entries in Table 1. These observations indicate that a lower training loss under DropoutTS does not transfer to unseen windows: the dropped timesteps are not selected by recoverability or forecasting harm, so the backbone overfits local fluctuations. In contrast, Ours estimates which patches can be reconstructed by the diffusion head and which ones would raise the forecast loss if removed, then spends the mask budget on redundant history while keeping regime-changing patches. The resulting training signal is more conservative on the observed split but more accurate on the test split. Additionally, DropoutTS exhibits pronounced oscillations in the test MSE on some datasets (e.g., Figure 5(b) and (d)), whereas Ours remains stable after a few epochs. Since the mask is disabled at inference, this gain in generalization does not add test-time overhead.

H Visualization

I Qualitative visualization of the two-view mask

The plugin never changes the backbone at test time: the mask is a training-time view of the look-back window. To make that construction concrete, Figure 7 shows one ILI window with the same geometry as the main protocol (L=36, H=24, patch length 16, stride 8). A single Top-k mask is

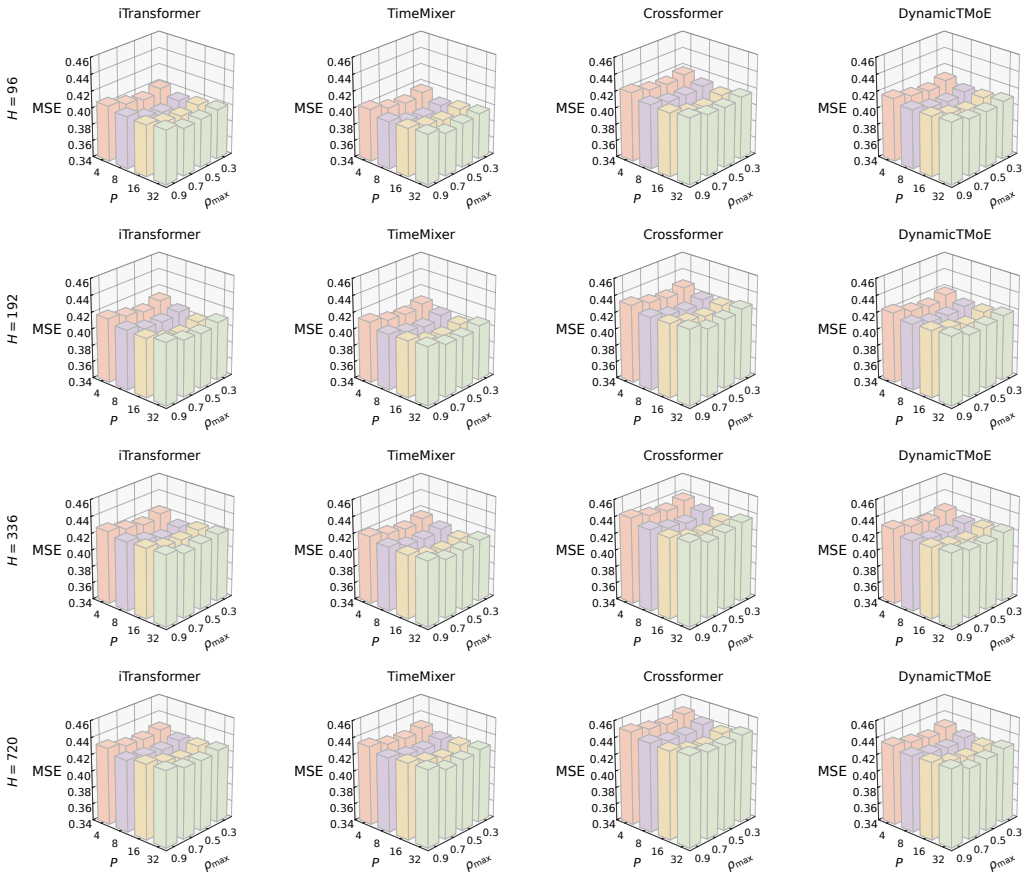


Figure 3: Dual-parameter sensitivity of DiffusionMask to the number of look-back patches P and the maximum mask budget $\rho_{\max}$. Each row is one forecast horizon $H \in \{96, 192, 336, 720\}$; each column is one backbone (iTransformer, TimeMixer, Crossformer, DynamicTMoE). In every panel the floor axes are $P \in \{4, 8, 16, 32\}$ (left) and $\rho_{\max} \in \{0.3, 0.5, 0.7, 0.9\}$ (right); bars of the same P share a color. The vertical axis is test MSE, zoomed to $[0.30, 0.46]$ so the $\sim$0.01 gaps are visible. The look-back length is L=96. Lower is better.

applied to the look-back patches and is then reused in two places. On the forecast path, masked patches are replaced by a learned mask token, so the backbone sees a gapped history (the gold curve in Figure 7(a) peels away from the true series only on the dropped patch). On the diffusion path, the same patches are the only ones that receive the forward noising process of the recon head (Figure 7(b)): visible patches stay clean, while the dropped patch is driven from a noisy state at t=4 toward the clean patch at t=4. The recon head is trained to predict the injected noise on those dropped patches, and the resulting recoverability scores decide which patches can spend the mask budget.

Figure 8 restates this construction as a patch unfold. Each heatmap is one look-back reshaped into a $P \times S$ map. The red box marks the dropped patch and does not move across rows: the forecasting view fills that row with a mask token, and the reconstruction view applies forward diffusion only there. The three columns slide the window by one patch so that the box tracks the same span; they are not a second masking rule. Because the mask is disabled at inference, the two-view procedure adds no test-time overhead.

Table 7: Efficiency analysis on ETTm1 ($L = 336$, $H = 96$). Methods use the same batch size and 40 training epochs. Training time includes four audit rounds for `DIAMOND`; inference uses the full unmasked input.

Backbone	Method	Step (ms/iter)	Audit (s/round)	Total (min)	Peak GPU (MiB)	Inference (ms/batch)	MSE ↓
Crossformer	Vanilla	92	–	32.75	2780	13.0	0.334
	Random Mask	94	–	33.46	2790	13.0	0.325
	DropoutTS	102	–	36.31	2890	13.0	0.319
	`DIAMOND`	178	35	65.70	4300	13.0	0.306
TimesNet	Vanilla	78	–	27.77	2320	10.5	0.326
	Random Mask	80	–	28.48	2330	10.5	0.317
	DropoutTS	89	–	31.68	2440	10.5	0.315
	`DIAMOND`	151	32	55.89	3720	10.5	0.301
iTransformer	Vanilla	49	–	17.44	1450	6.9	0.308
	Random Mask	50	–	17.80	1460	6.9	0.302
	DropoutTS	55	–	19.58	1510	6.9	0.296
	`DIAMOND`	106	24	39.34	2580	6.9	0.284
TimeMixer	Vanilla	43	–	15.31	1340	5.8	0.313
	Random Mask	44	–	15.66	1350	5.8	0.307
	DropoutTS	51	–	18.16	1410	5.8	0.301
	`DIAMOND`	103	22	38.13	2470	5.8	0.293
Dynamic TMoE	Vanilla	118	–	42.01	3140	14.5	0.298
	Random Mask	120	–	42.72	3150	14.5	0.295
	DropoutTS	132	–	46.99	3270	14.5	0.289
	`DIAMOND`	209	38	76.94	4880	14.5	0.277
PatchTST	Vanilla	55	–	19.58	1640	7.8	0.301
	Random Mask	56	–	19.94	1650	7.8	0.294
	DropoutTS	62	–	22.07	1720	7.8	0.297
	`DIAMOND`	119	26	44.10	2890	7.8	0.280

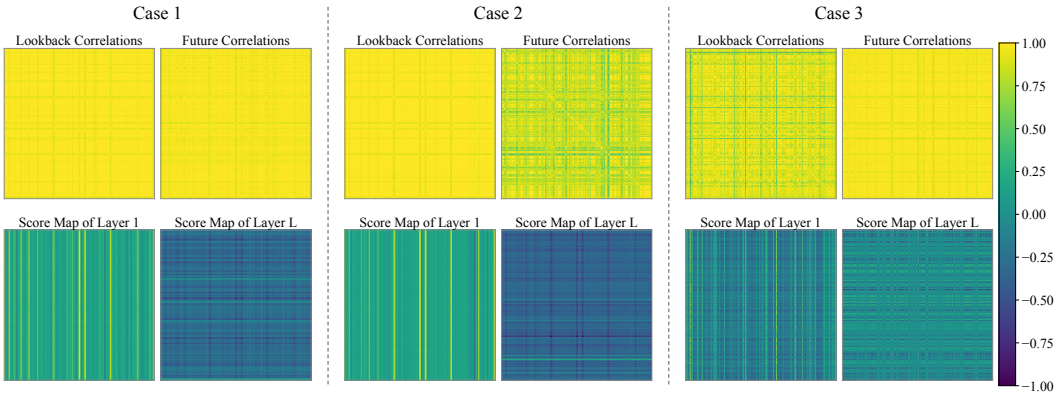


Figure 4: Qualitative Solar cases (C=137 stations). **Top:** pairwise Pearson correlations on the look-back window (L=96) and the forecast horizon (H=192). Case 1 is day→day, Case 2 day→night, and Case 3 night→day. **Bottom:** head-averaged pre-softmax attention of the inverted Transformer encoder (Layer 1 / Layer L) on the same look-back windows. Daytime maps are nearly uniformly correlated; crossing dusk or dawn sparsifies the future correlations. Encoder attention is column-sparse rather than a copy of the Pearson maps.

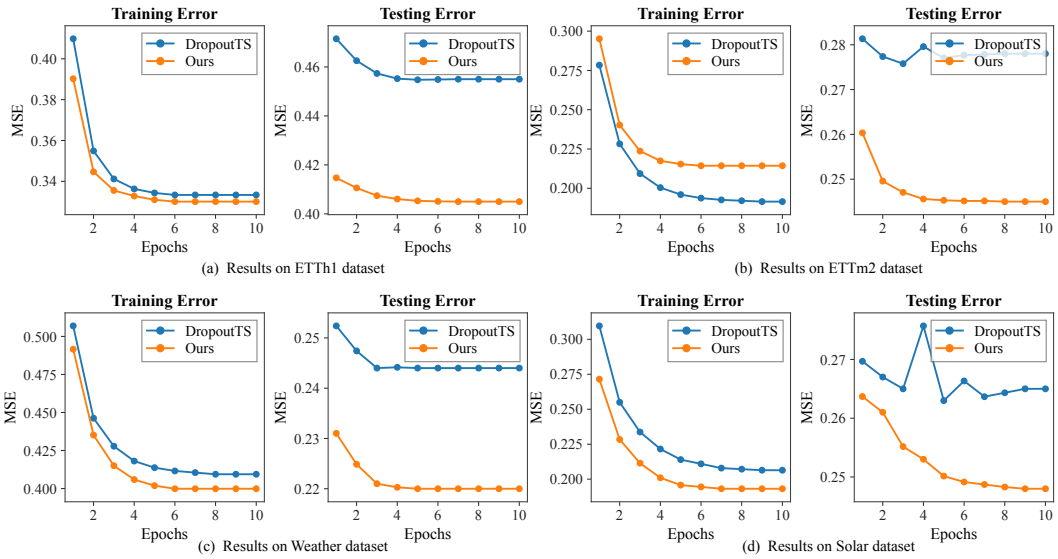


Figure 5: Training and testing MSE of DropoutTS and Ours across epochs on (a) ETTh1, (b) ETTm2, (c) Weather, and (d) Solar. Left: training error. Right: testing error.

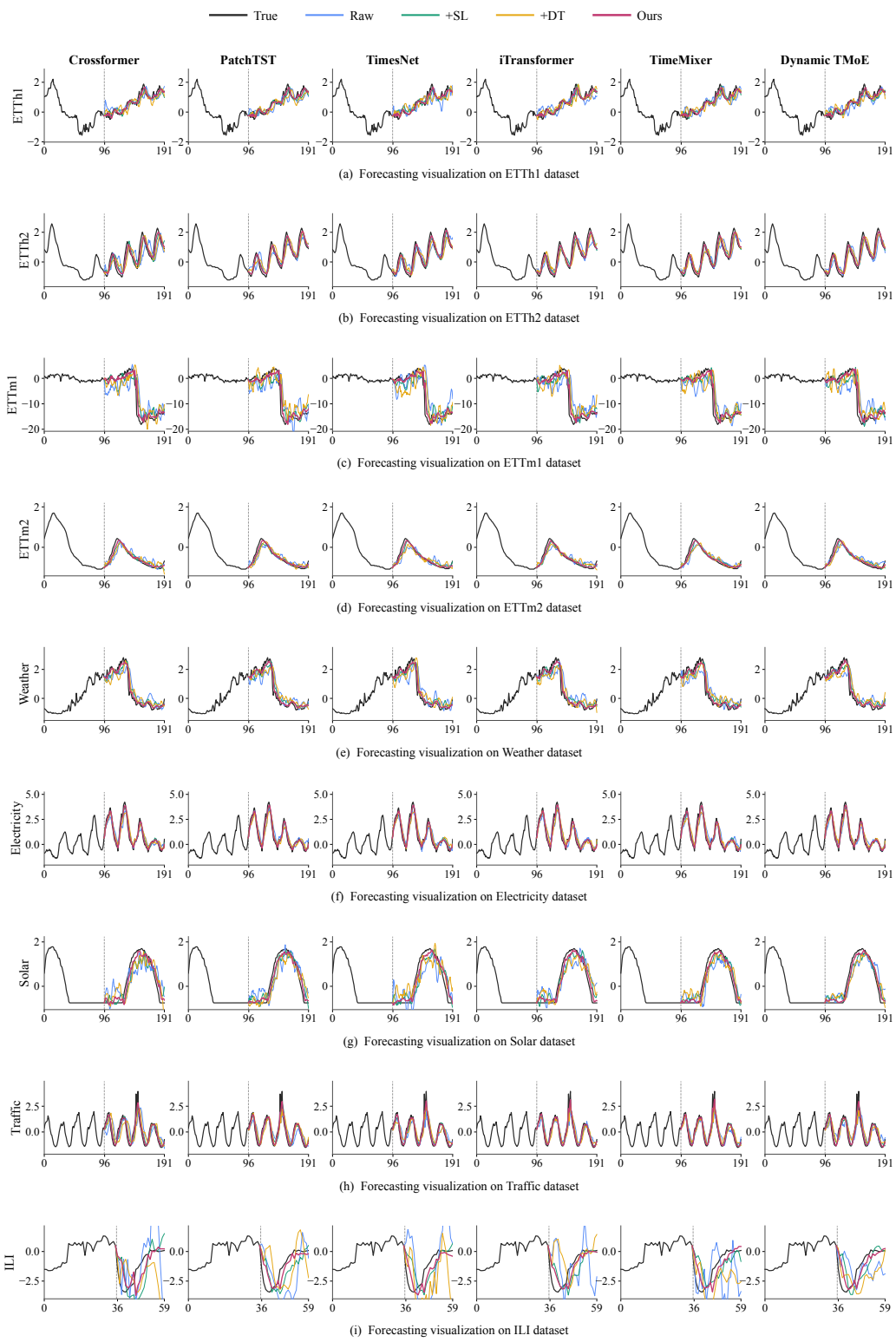


Figure 6: Forecasting visualization on all datasets. Rows are (a) ETTh1, (b) ETTh2, (c) ETTm1, (d) ETTm2, (e) Weather, (f) Electricity, (g) Solar, (h) Traffic, and (i) ILI. Columns are Crossformer, PatchTST, TimesNet, iTransformer, TimeMixer, and Dynamic TMoE. Black is the ground-truth series on the look-back window and the horizon; the dashed line marks the forecast start. Colored curves are Raw, Selective Learning (+SL), DropoutTS (+DT), and Ours on the horizon. The input length is L=96 and the horizon is H=96, except ILI (L=36, H=24).

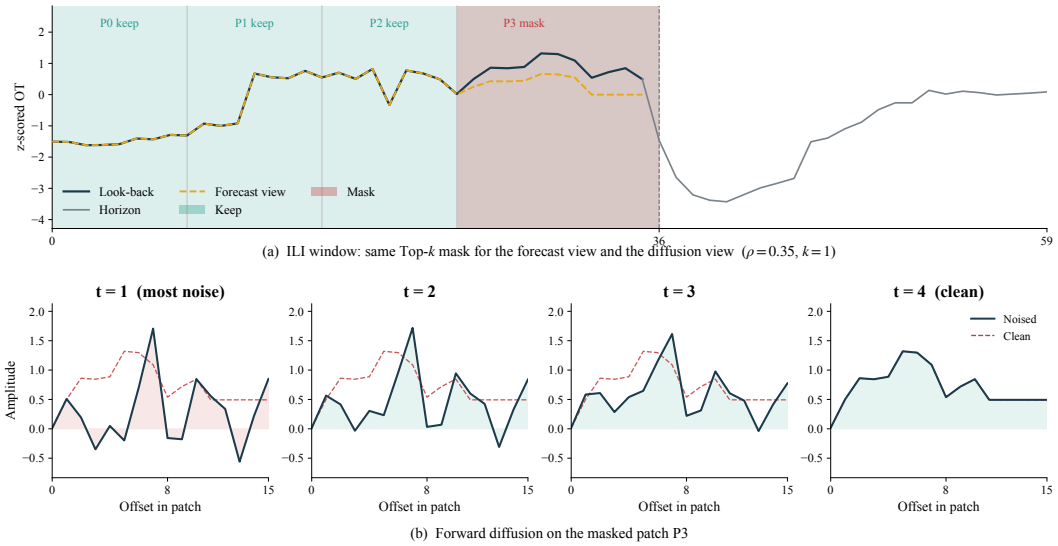


Figure 7: Training-time views of DiffusionMask on one ILI window (L=36, H=24). **(a)** Look-back patches under a Top-k mask with ρ=0.35 (k=1). Teal spans are kept; the red span is masked. The gold dashed curve is the forecast view after replacing the masked patch with a mask token; the gray curve is the horizon after the dashed split. **(b)** Forward diffusion on the masked patch P3. Only this patch is noised; t=1 is the noisiest step and t=4 recovers the clean patch under the recon-head schedule.

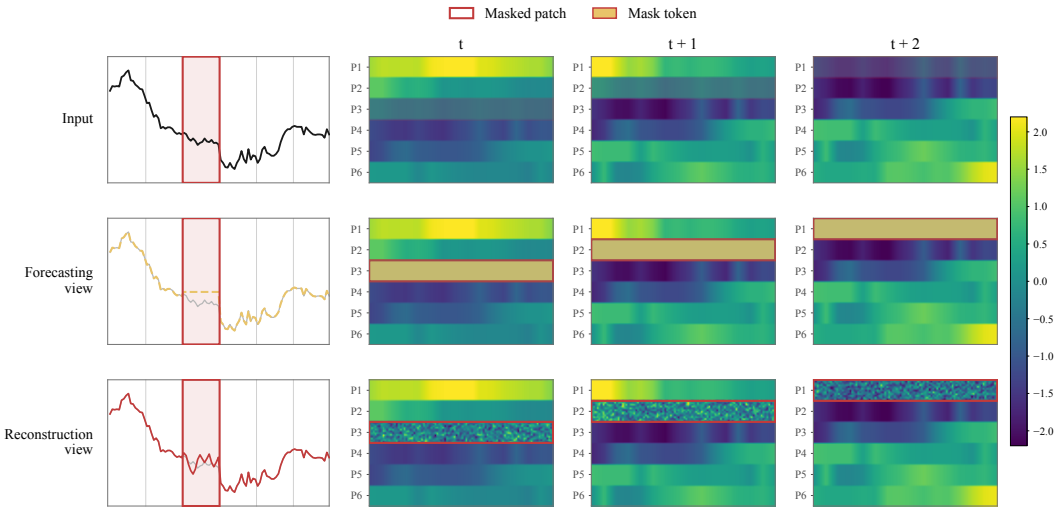


Figure 8: The same two-view mask after unfolding the look-back into patches, shown on sliding ETTh1 windows. Rows are the input, the forecasting view, and the diffusion reconstruction view; columns advance the window by one patch. The red box is the masked patch and is shared by both views. Gold is replacement by a learned mask token; speckle inside the box is forward diffusion on that patch only.