TIMEGDC: TOWARDS EFFICIENT AND ROBUST JOINT DISTRIBUTION LEARNING FOR LONG-TERM PROBABILISTIC TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Traditional point-based time series forecasting provides limited information for risk-aware decision-making. Long-term probabilistic forecasting methods, however, often suffer from low computational efficiency and limited robustness, especially when dealing with complex distributions and potentially contaminated observations over extended forecast horizons. To address these challenges, we propose **TimeGDC**, a contamination-aware framework for efficient joint probabilistic forecasting over long horizons. TimeGDC jointly models the predictive distribution over the entire forecasting horizon using a ***G**aussian mixture distribution* that captures multimodal uncertainty. Each mixture component is parameterized with a ***D**ynamically smoothed low-rank covariance* structure to efficiently represent temporal dependencies among future time steps. Furthermore, we construct a ***C**ontamination-aware likelihood* for joint forecasting by combining the predictive distribution with a fixed heavy-tailed reference distribution, thereby reducing the influence of abnormal observations on distribution learning. Extensive experiments on six widely used multivariate time-series benchmark datasets show that TimeGDC consistently outperforms four competitive probabilistic forecasting baselines, achieving average relative reductions of 16.6%, 7.7%, and 13.4% in CRPS, NMAE, and Energy Score, respectively.

1 INTRODUCTION

In recent years, time series forecasting has been widely applied in energy, transportation, finance, and industrial applications. Sun et al. (2022); Ermagun & Levinson (2018); Sezer et al. (2020). For example, electricity load forecasting supports power dispatch, traffic forecasting helps transportation planning, and financial forecasting provides useful information for risk control.

Point forecasting produces a single value for each future time step but does not describe prediction uncertainty. Probabilistic time series forecasting (PTSF) instead estimates the conditional distribution of future observations given historical data Gneiting & Katzfuss (2014). Existing likelihood-based methods can be trained efficiently but often rely on simple distribution assumptions Salinas et al. (2020; 2019); Rangapuram et al. (2018); Rasul et al. (2020); Drouin et al. (2022). In addition, potentially contaminated observations, i.e., observations containing abnormal values that deviate from the predictive distribution, may disturb likelihood-based distribution learning. Generative methods provide more flexible distributions but usually require repeated sampling Rasul et al. (2021); Tashiro et al. (2021); Alcaraz & Strodthoff (2022); Shen & Kwok (2023); Kollovieh et al. (2023); Li et al. (2024).

These issues are more pronounced in long-term probabilistic time series forecasting (LPTSF) Bergsma et al. (2023); Tang & Matteson (2021); Le Guen & Thome (2020); Wu et al. (2025). Two challenges are central: efficient joint distribution learning over long horizons and robust learning from potentially contaminated observations. First, assuming independent future steps ignores their correlations, while autoregressive prediction requires sequential estimation. A full covariance captures inter-step correlations but has high parameter and matrix costs and may introduce noise. Second, long-horizon predictive distributions can be multimodal, heavy-tailed, and

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.

heteroskedastic. Simple parametric distributions may not capture this complexity, and potentially contaminated observations can further disrupt distribution learning.

To address these issues, we propose TimeGDC, a robust probabilistic framework for long-term time series forecasting. TimeGDC consists of three key components: a Gaussian mixture distribution that jointly models each channel's entire forecasting horizon to capture multimodal uncertainty, a dynamically smoothed low-rank covariance structure that efficiently models inter-step correlations, and a contamination-aware likelihood with a fixed heavy-tailed reference distribution that reduces the influence of potentially contaminated observations during training.

The main contributions of this paper are as follows.

- We propose an efficient joint probabilistic forecasting framework that models the entire forecasting window with structured Gaussian mixture distributions and a dynamically smoothed low-rank covariance structure, enabling multimodal uncertainty modeling and inter-step correlation modeling while avoiding the high computational cost of full covariance estimation.
- We develop a robust distribution learning approach for long-horizon forecasting by introducing a contamination-aware likelihood with a fixed heavy-tailed reference distribution, which reduces the influence of potentially contaminated observations during training.
- Extensive experiments on six widely used multivariate time-series benchmark datasets demonstrate the effectiveness of TimeGDC, achieving superior probabilistic forecasting performance and improved robustness under controlled contamination settings compared with competitive baselines.

2 Related Work

2.1 Long-term Time Series Forecasting

Early long-term forecasting used statistical models such as ARIMA Box et al. (2015). Recurrent, convolutional, and Transformer models later captured nonlinear dependencies, while sparse attention, decomposition, frequency modeling, patching, and channel-independent modeling improved long-horizon forecasts Zhou et al. (2021); Wu et al. (2021); Zhou et al. (2022); Nie et al.; Liu et al. (2024). However, most of these methods focus on point forecasts and do not explicitly characterize predictive uncertainty.

2.2 Probabilistic Forecasting Methods

Probabilistic forecasting methods characterize future uncertainty by estimating predictive distributions rather than single future values. Traditional methods use distributional assumptions or resampling, such as Bootstrap Efron (1992). DeepAR learns predictive distributions autoregressively, while GPVar extends this approach to multivariate forecasting Salinas et al. (2020; 2019). State-space models, normalizing flows, and copulas further improve distribution flexibility and dependency modeling Rangapuram et al. (2018); Rasul et al. (2020); Drouin et al. (2022). However, predefined distributions or autoregressive factorization may limit multimodal modeling and joint dependencies across future steps. Sampling-based methods include autoregressive, conditional score-based, and non-autoregressive diffusion models Rasul et al. (2021); Tashiro et al. (2021); Alcaraz & Strodthoff (2022); Shen & Kwok (2023); Kollovieh et al. (2023); Li et al. (2024), as well as ensembles and MC Dropout Gal & Ghahramani (2016). They offer flexible predictive distributions but require repeated sampling or denoising, increasing inference time and memory use.

2.3 Long-term Probabilistic Time Series Forecasting

Recent studies further explore probabilistic forecasting under long-sequence and long-horizon settings. SutraNets uses sub-series autoregression to reduce the difficulty of modeling long future sequences Bergsma et al. (2023). Probabilistic Transformer introduces a non-autoregressive probabilistic forecasting framework, while STRIPE encourages diverse predictions from both shape and temporal perspectives Tang & Matteson (2021); Le Guen & Thome (2020). K^2VAE combines variational modeling with Koopman–Kalman latent dynamics to reduce error accumulation in long-term

forecasting Wu et al. (2025). Although these methods improve the flexibility of long-term probabilistic forecasting, they may still face error accumulation, high sampling costs, insufficient explicit temporal dependency modeling, and limited robustness to potentially contaminated observations.

3 METHOD

In this paper, we propose TimeGDC, a probabilistic framework that combines **G**aussian mixture-based joint distribution modeling, **D**ynamically smoothed low-rank covariance, and a **C**ontamination-aware likelihood for long-horizon forecasting. The Gaussian mixture models the entire forecasting window jointly to capture multimodal uncertainty. For each component, diagonal and low-rank covariance terms, together with a learnable temporal persistence constraint, efficiently model dependencies among future time steps. During training, the contamination-aware likelihood uses a fixed heavy-tailed reference distribution to reduce the gradient contribution of potentially contaminated observations. The overall framework is illustrated in Figure 1.

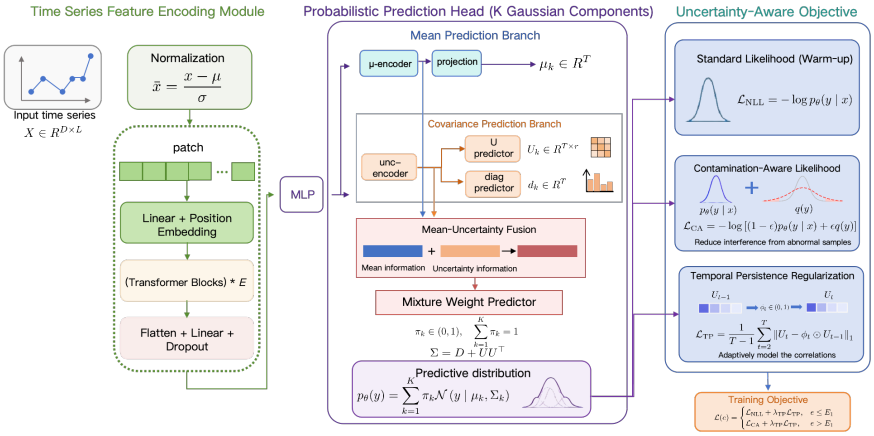

Figure 1: **Overall architecture of the proposed method.** *Left*: A time series encoding module performs normalization, patching, positional encoding, and E Transformer layers to extract temporal representations. *Middle*: A probabilistic prediction head estimates the means and diagonal-plus-low-rank covariances of K Gaussian components through separate mean and covariance branches, and predicts mixture weights by fusing their representations. *Right*: The uncertainty-aware objective consists of a contamination-aware likelihood that reduces the influence of potentially contaminated observations and a temporal persistence regularization term that stabilizes covariance learning.

3.1 PROBLEM FORMULATION

We first define the long-term probabilistic time series forecasting (LPTSF) problem. Under the rolling forecasting setting with a fixed-size window, at time t, the historical input sequence is denoted as

$$\mathbf{X}^t = \left\{ \mathbf{x}_1^t, \mathbf{x}_2^t, \ldots, \mathbf{x}_L^t \mid \mathbf{x}_i^t \in \mathbb{R}^{C_x} \right\}, \tag{1}$$

where L is the input window length and C_x is the number of input feature channels. The forecasting target is the future sequence

$$\mathbf{Y}^t = \left\{ \mathbf{y}_1^t, \mathbf{y}_2^t, \ldots, \mathbf{y}_T^t \mid \mathbf{y}_i^t \in \mathbb{R}^{C_y} \right\}, \tag{2}$$

where T denotes the prediction horizon and C_y is the number of output channels. We formulate probabilistic forecasting as conditional density estimation and learn $p_\theta(\mathbf{Y}^t \mid \mathbf{X}^t)$ by maximizing the conditional log-likelihood of the observed training samples:

$$\theta^\star = \arg\max_\theta \sum_t \log p_\theta(\mathbf{Y}^t \mid \mathbf{X}^t). \tag{3}$$

3.2 JOINT PROBABILISTIC MODELING ON TIME SERIES

Many explicit probabilistic forecasting methods assume a predefined conditional distribution, such as a Gaussian, and learn its parameters from historical observations Salinas et al. (2020). Although efficient, a single Gaussian may not adequately capture multimodality, heavy tails, heteroscedasticity, and abrupt changes in real-world time series. We therefore use a Gaussian mixture to model more complex predictive distributions while retaining likelihood-based learning Pearson (1895). Following evidence that channel-independent modeling is effective and stable for multivariate forecasting Han et al. (2024), we forecast each channel independently with parameters shared across channels.

Different from autoregressive methods that estimate future distributions step by step Salinas et al. (2020), we model each channel's entire forecasting window as a multivariate joint random variable and predict it non-autoregressively, avoiding repeated distribution estimation and potential error accumulation. Each mixture component has a mean vector and an explicit covariance matrix across future steps, allowing the model to assign probability mass to different temporal modes while capturing inter-step correlations.

In summary, for a single-channel input window $\mathbf{x} \in \mathbb{R}^L$ and the corresponding forecasting window $\mathbf{y} \in \mathbb{R}^T$, the conditional joint density function can be written as:

$$p(\mathbf{y} \mid \mathbf{x}) = \sum_{j=1}^{K} \pi_j \, \mathcal{N}(\mathbf{y}; \mu_j, \Sigma_j), \tag{4}$$

where

$$\mathcal{N}(\mathbf{y}; \mu_j, \Sigma_j) = (2\pi)^{-\frac{T}{2}} |\Sigma_j|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (\mathbf{y} - \mu_j)^\top \Sigma_j^{-1} (\mathbf{y} - \mu_j) \right). \tag{5}$$

During training, we use the negative log-likelihood loss as the optimization objective and jointly learn the mixture weights, mean parameters, and covariance structure. The corresponding loss function $\mathcal{L}_{\mathrm{NLL}}(\mathbf{x}, \mathbf{y})$ can be written as:

$$\mathcal{L}_{\mathrm{NLL}}(\mathbf{x}, \mathbf{y}) = \frac{T}{2} \log(2\pi) - \log \sum_{j=1}^{K} \pi_j |\Sigma_j|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (\mathbf{y} - \mu_j)^\top \Sigma_j^{-1} (\mathbf{y} - \mu_j) \right). \tag{6}$$

3.3 TIME SERIES FEATURE ENCODING

For an input time series $\mathbf{X} \in \mathbb{R}^{C \times L}$, each channel is normalized as $\tilde{x}_t^{(c)} = (x_t^{(c)} - \mu_c)/\sigma_c$ using training-set statistics, which are also used during testing. Following patch-based representation learning Nie et al., each channel is divided into N patches of length P and stride S. The patches are projected into d-dimensional embeddings, combined with positional encodings, and processed by an E-layer Transformer encoder Ashish (2017). The resulting representation $H^{(E,c)} \in \mathbb{R}^{N \times d}$ is flattened and projected to predict the Gaussian mixture parameters.

3.4 PROBABILISTIC PREDICTION HEAD

Based on the temporal context representation, the probabilistic prediction head produces the parameters of a K-component Gaussian mixture. For each component j, the mean branch predicts a mean vector $\mu_j \in \mathbb{R}^T$, while the covariance branch predicts a diagonal vector $d_j \in \mathbb{R}^T$ and a low-rank representation $U_j \in \mathbb{R}^{T \times r}$. The features from the two branches are then fused to predict the mixture weights $\{\pi_j\}_{j=1}^K$, where $\pi_j \geq 0$ and $\sum_{j=1}^K \pi_j = 1$. Therefore, each mixture weight depends on both the predicted future trend and its uncertainty, allowing the model to assign probability mass to different future patterns. The resulting parameters $\{\pi_j, \mu_j, d_j, U_j\}_{j=1}^K$ define the joint predictive distribution described in Section 3.2.

3.5 EFFICIENT AND ROBUST JOINT DISTRIBUTION LEARNING

After the prediction head outputs the Gaussian mixture parameters, we define the structured covariance and its temporal persistence regularizer. We then introduce the contamination-aware likelihood and combine it with the regularizer in the two-stage training objective.

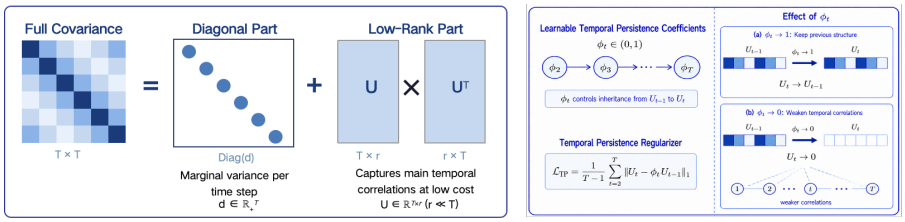

Figure 2: **Dynamic temporal smoothing of low-rank covariance.** Left: each covariance is decomposed into diagonal and low-rank terms. Right: learnable scalar coefficients control temporal persistence between adjacent low-rank factors.

Dynamic Temporal Smoothing of Low-Rank Covariance (DTSLC). A full covariance matrix can model correlations among future time steps, but requires many parameters and costly matrix operations, and may introduce redundant noise. In contrast, a diagonal covariance matrix is efficient but cannot model correlations between different time steps. To balance dependency modeling and efficiency, we decompose the covariance matrix of each Gaussian component into diagonal and low-rank parts, as illustrated in the left panel of Figure 2.

For Gaussian component j, the covariance matrix is:

$$\Sigma_j = \mathrm{Diag}(d_j) + U_j U_j^\top. \tag{7}$$

Here $d_j \in \mathbb{R}_+^T$ models marginal variances, and $U_j \in \mathbb{R}^{T\times r}$ captures correlations across future steps, where r is the low-rank dimension.

A full covariance matrix requires $\mathcal{O}(T^2)$ parameters for each mixture component, whereas the proposed diagonal-plus-low-rank structure requires only $T(r+1)$ covariance parameters, corresponding to $\mathcal{O}(Tr)$ when $r \ll T$. We evaluate the covariance terms without forming Σ_j^{-1}. Let $D_j = \mathrm{Diag}(d_j)$, $M_j = I_r + U_j^\top D_j^{-1} U_j$, and $q_j = U_j^\top D_j^{-1}(\mathbf{y} - \mu_j)$. Using the Woodbury identity and matrix determinant lemma,

$$(\mathbf{y}-\mu_j)^\top \Sigma_j^{-1}(\mathbf{y}-\mu_j) = (\mathbf{y}-\mu_j)^\top D_j^{-1}(\mathbf{y}-\mu_j) - q_j^\top M_j^{-1} q_j, \tag{8}$$

$$\log|\Sigma_j| = \sum_{t=1}^{T} \log d_{j,t} + \log|M_j|. \tag{9}$$

Cholesky factorization is applied only to the $r \times r$ matrix M_j, yielding $\mathcal{O}(Tr^2 + r^3)$ time per component instead of $\mathcal{O}(T^3)$.

The low-rank matrix captures the main covariance patterns with few parameters, but unconstrained optimization may produce unstable temporal representations. Since adjacent time steps usually exhibit local smoothness Kitagawa & Gersch (1996), we apply a temporal persistence regularizer with learnable coefficients to the low-rank representation. The temporal persistence regularizer $\mathcal{L}_{\mathrm{TP}}$ is defined as follows:

$$\mathcal{L}_{\mathrm{TP}} = \frac{1}{T-1}\sum_{t=2}^{T} \|U_t - \phi_t U_{t-1}\|_1, \tag{10}$$

where $\phi_t = \mathrm{sigmoid}(\theta_t)$ ensures $\phi_t \in (0,1)$, and $\boldsymbol{\phi} = (\phi_2, \ldots, \phi_T)$ is the learnable temporal persistence vector. Each element ϕ_t is a scalar coefficient that controls the persistence between two adjacent prediction steps. When $\phi_t \to 1$, the regularizer encourages $U_t \to U_{t-1}$, so that the current time step inherits the low-rank covariance structure of the preceding time step. When $\phi_t \to 0$, the regularizer shrinks U_t toward zero, thereby weakening the correlations between time step t and the other time steps. The effect of ϕ_t is illustrated in the right panel of Figure 2.

Contamination-Aware Likelihood. Following contamination-mixture modeling in robust statistics, we combine the window-level Gaussian mixture predictive density with a fixed multivariate Student-t reference density. For the standardized target $\mathbf{y} \in \mathbb{R}^T$, the reference density is:

$$q(\mathbf{y}) = \frac{\Gamma\left(\frac{\nu+T}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)(\nu\pi)^{T/2} s^T}\left(1 + \frac{\|\mathbf{y}\|_2^2}{\nu s^2}\right)^{-\frac{\nu+T}{2}}, \tag{11}$$

where ν and s are the degrees of freedom and isotropic scale parameter, respectively. Both parameters are fixed during training, so q is independent of the model parameters.

Using this reference density, we define the contamination-aware observation density used for training as

$$p_{\text{obs}}(\mathbf{y} \mid \mathbf{x}) = (1-\epsilon)p_\theta(\mathbf{y} \mid \mathbf{x}) + \epsilon q(\mathbf{y}), \tag{12}$$

where $\epsilon \in [0, 1)$ is a fixed mixing coefficient representing the prior probability of contamination. The corresponding contamination-aware negative log-likelihood is

$$\mathcal{L}_{\text{CA}}(\mathbf{x}, \mathbf{y}) = -\log\left[(1-\epsilon)p_\theta(\mathbf{y} \mid \mathbf{x}) + \epsilon q(\mathbf{y})\right]. \tag{13}$$

Because both q and ϵ are fixed and independent of θ, the gradient of this loss with respect to the forecasting model is

$$\nabla_\theta \mathcal{L}_{\text{CA}} = -w(\mathbf{x}, \mathbf{y}) \nabla_\theta \log p_\theta(\mathbf{y} \mid \mathbf{x}), \tag{14}$$

where

$$w(\mathbf{x}, \mathbf{y}) = \frac{(1-\epsilon)p_\theta(\mathbf{y} \mid \mathbf{x})}{(1-\epsilon)p_\theta(\mathbf{y} \mid \mathbf{x}) + \epsilon q(\mathbf{y})}. \tag{15}$$

The weight approaches one when the predictive density dominates and decreases when the reference density dominates, reducing the influence of potentially contaminated samples. At inference time, only $p_\theta(\mathbf{y} \mid \mathbf{x})$ is used.

Overall Training Objective. The temporal persistence regularizer is applied throughout training, while the likelihood term changes after the warm-up stage. Therefore, the overall training objective at epoch e is defined as

$$\mathcal{L}(e) = \begin{cases} \mathcal{L}_{\text{NLL}} + \lambda_{\text{TP}} \mathcal{L}_{\text{TP}}, & 1 \le e \le E_1, \\ \mathcal{L}_{\text{CA}} + \lambda_{\text{TP}} \mathcal{L}_{\text{TP}}, & E_1 < e \le E_2, \end{cases} \tag{16}$$

where λ_{TP} controls the contribution of the temporal persistence regularizer.

4 Experimental Evaluation

4.1 Dataset Settings

We evaluate all methods on six real-world multivariate benchmarks: Electricity, ETTh1, ETTh2, ETTm1, ETTm2, and Weather Zhou et al. (2021); Wu et al. (2021), using an input length of $L = 96$ and forecasting horizons $T \in \{96, 192, 336, 720\}$. Detailed dataset statistics and train/validation/test splits are provided in Appendix A.

4.2 Evaluation Metrics

We evaluate marginal distribution quality using the Continuous Ranked Probability Score (CRPS) Matheson & Winkler (1976); Gneiting & Raftery (2007), overall joint distribution quality using the Energy Score (ES) Gneiting & Raftery (2007), and point prediction accuracy using NMAE Wu et al. (2025). Following the CSDI evaluation protocol Tashiro et al. (2021), CRPS is approximated using quantile levels $\{0.05, 0.10, \ldots, 0.95\}$ and globally normalized over all test windows, target channels, and forecasting steps, while ES is estimated from $S = 100$ predictive samples. Lower values indicate better performance for all metrics. Detailed definitions are provided in Appendix B.

4.3 Implementation Details

General Experimental Settings. All experiments follow the channel-independent multivariate forecasting setting, where each target channel is predicted from its history with parameters shared across channels. All models are trained with Adam using a batch size of 32 on an NVIDIA RTX 4090 GPU. CRPS and ES are computed directly from predictive samples, whereas NMAE is computed from point predictions.

Key Hyperparameter Configurations. The low-rank dimension is set to $r = 4$, the temporal persistence regularization weight is set to $\lambda_{\text{TP}} = 0.5$, and the persistence coefficients ϕ_t are initialized

Table 1: Performance comparison of CRPS, NMAE, and Energy Score (ES) across six datasets and four prediction lengths. Lower values indicate better performance. Best results are shown in bold, and second-best results are underlined. A dash denotes an out-of-memory (OOM) run. Averages and rankings are calculated from unrounded values.

Dataset	Pred Len	OURS			SIMDIFF(2026)			NSDiff(2025)			K2VAE(2025)			TACTiS-2(2024)		
		CRPS	NMAE	ES	CRPS	NMAE	ES	CRPS	NMAE	ES	CRPS	NMAE	ES	CRPS	NMAE	ES
Electricity	96	**0.235**	**0.254**	**2.457**	0.322	0.277	3.479	0.359	0.292	3.539	0.331	0.356	3.231	0.269	0.288	2.858
	192	**0.245**	**0.264**	**3.653**	0.322	0.276	5.041	0.364	0.303	5.100	0.346	0.371	4.820	0.315	0.342	4.637
	336	**0.260**	**0.281**	**5.116**	0.345	0.296	7.082	0.367	0.308	6.810	0.349	0.375	6.475	-	-	-
	720	**0.287**	**0.311**	**8.169**	0.378	0.322	11.364	0.391	0.340	10.583	0.356	0.382	9.648	-	-	-
	avg	**0.257**	**0.277**	**4.849**	0.342	0.293	6.741	0.370	0.311	6.508	0.345	0.371	6.043	-	-	-
ETTh1	96	**0.395**	**0.445**	**4.176**	0.609	0.542	6.401	0.473	0.504	4.763	0.588	0.653	5.960	0.618	0.669	6.243
	192	**0.440**	**0.497**	**6.541**	0.686	0.607	10.292	0.498	0.534	6.995	0.655	0.710	9.307	0.634	0.686	9.175
	336	**0.455**	**0.512**	**8.899**	0.695	0.626	13.911	0.568	0.612	10.335	0.713	0.732	13.594	0.655	0.716	12.318
	720	**0.531**	**0.597**	**14.961**	0.905	0.771	25.965	0.619	0.650	17.020	0.686	0.727	18.899	0.721	0.788	19.539
	avg	**0.455**	**0.512**	**8.644**	0.724	0.637	14.142	0.539	0.575	9.778	0.660	0.706	11.940	0.657	0.715	11.819
ETTh2	96	**0.271**	**0.283**	**2.466**	0.402	0.329	3.735	0.326	0.328	2.953	0.435	0.431	3.806	0.322	0.333	2.904
	192	**0.301**	**0.306**	**3.838**	0.408	0.332	5.372	0.356	0.355	4.527	0.576	0.557	6.852	0.347	0.353	4.416
	336	**0.322**	**0.328**	**5.388**	0.455	0.358	7.899	0.360	0.356	6.067	0.680	0.639	10.535	0.371	0.369	6.435
	720	**0.367**	**0.367**	**8.897**	0.479	0.392	11.956	0.410	0.399	10.027	0.655	0.626	14.984	0.448	0.441	11.545
	avg	**0.315**	**0.321**	**5.147**	0.436	0.353	7.240	0.363	0.359	5.894	0.587	0.563	9.045	0.372	0.374	6.325
ETTm1	96	**0.376**	0.406	**3.937**	0.445	**0.397**	4.713	0.474	0.493	4.645	0.459	0.492	4.635	0.404	0.438	4.383
	192	**0.399**	0.436	**5.914**	0.495	**0.436**	7.543	0.476	0.501	6.564	0.509	0.533	7.327	0.442	0.485	6.693
	336	**0.432**	**0.470**	**8.406**	0.553	0.482	11.139	0.495	0.518	9.225	0.536	0.575	10.155	0.611	0.645	10.569
	720	**0.462**	**0.508**	**13.175**	0.612	0.531	17.866	0.570	0.591	15.332	0.593	0.618	16.509	0.631	0.678	17.417
	avg	**0.417**	**0.455**	**7.858**	0.526	0.462	10.315	0.504	0.526	8.942	0.524	0.554	9.656	0.522	0.561	9.766
ETTm2	96	**0.221**	**0.228**	**2.067**	0.317	0.269	2.985	0.258	0.249	2.371	0.315	0.325	2.800	0.240	0.248	2.335
	192	**0.251**	**0.258**	**3.323**	0.335	0.279	4.521	0.281	0.274	3.664	0.399	0.401	5.042	0.307	0.311	3.897
	336	**0.286**	**0.295**	**4.965**	0.347	0.303	6.159	0.309	0.305	5.345	0.446	0.447	7.343	0.311	0.319	5.497
	720	**0.325**	**0.331**	**8.098**	0.434	0.349	11.281	0.357	0.352	8.889	0.488	0.498	11.819	0.352	0.359	8.860
	avg	**0.271**	**0.278**	**4.613**	0.358	0.300	6.236	0.301	0.295	5.067	0.412	0.418	6.751	0.303	0.309	5.147
Weather	96	0.271	0.221	**2.136**	0.434	0.283	3.321	0.356	0.264	2.684	0.348	0.283	2.591	**0.258**	**0.215**	2.170
	192	**0.323**	**0.264**	**3.652**	0.468	0.327	5.161	0.408	0.315	4.363	0.407	0.330	4.368	0.344	0.288	4.059
	336	**0.378**	**0.307**	**5.604**	0.531	0.346	8.129	0.462	0.346	6.559	0.456	0.375	6.491	0.422	0.345	6.706
	720	**0.456**	**0.373**	**9.827**	0.622	0.433	13.795	0.539	0.418	11.507	0.550	0.452	11.524	-	-	-
	avg	**0.357**	**0.291**	**5.305**	0.514	0.347	7.602	0.441	0.336	6.278	0.440	0.360	6.243	-	-	-

to 0.6. Two-stage training uses $E_1 = 7$ and $E_2 = 10$, with the standard negative log-likelihood used in the first stage and the contamination-aware likelihood used in the second stage. We fix the contamination coefficient at $\epsilon = 0.05$ and set the multivariate Student-t reference distribution parameters to $\nu = 3$ and $s = 1$ whenever the contamination-aware likelihood is active. Before model training, we apply MiniBatchKMeans Sculley (2010) to the forecasting windows in the training set for different candidate values of K. We use a hybrid selection criterion: the elbow point of the BIC curve determines the upper bound of the candidate range, and the value with the highest silhouette score within this range is selected. The resulting K is then fixed throughout model training. The selected values of K are reported in Appendix C.

Progressive Low-Rank Covariance Optimization. To stabilize covariance learning, we freeze the low-rank covariance and temporal persistence parameters during epochs 1–4, while optimizing the diagonal covariance branch and the remaining parameters. From epoch 5, all parameters are jointly optimized, introducing temporal dependency modeling after stable marginal uncertainty estimation.

4.4 Comparative Experiments

We compare TimeGDC with four recent forecasting methods covering different modeling mechanisms. SimDiff uses a unified Transformer as both denoiser and predictor and applies median-of-means aggregation to repeated diffusion outputs for point forecasting Ding et al. (2026). For SimDiff, CRPS and ES are computed from 100 raw trajectories generated by independent conditional reverse-diffusion runs from Gaussian noise, whereas median-of-means aggregation is used only for NMAE. NSDiff combines a location–scale diffusion model with a pre-trained mean–variance estimator and an uncertainty-aware noise schedule to model non-stationary uncertainty Ye et al. (2025). K2VAE integrates Koopman-based linear latent dynamics and Kalman-based uncertainty refinement within a variational framework Wu et al. (2025). TACTiS-2 uses a transformer-based attentional copula to model multivariate predictive dependencies Ashok et al. (2024). Following common practice in long-term forecasting, all methods use the same fixed seed, data splits, input and forecasting lengths, and training and evaluation settings. The results are reported in Table 1.

Table 2: Training and inference time on different datasets and forecasting horizons. Lower values indicate better efficiency. Best results are shown in bold, and second-best results are underlined. A dash denotes an out-of-memory (OOM) run.

Dataset	Pred Len	OURS		SimDiff		NSDiff		K2VAE		TACTiS-2	
		Train (min)	Infer (s)	Train (min)	Infer (s)	Train (min)	Infer (s)	Train (min)	Infer (s)	Train (min)	Infer (s)
Electricity	96	10.22	**5.09**	23.30	5161.00	**4.48**	629.00	5.10	42.00	33.55	2983.00
	192	10.12	**8.29**	33.63	6791.00	**4.57**	655.00	6.37	80.00	95.89	5885.53
	336	10.38	**14.90**	47.20	9371.00	**4.70**	717.00	8.58	137.00	-	-
	720	16.63	**25.77**	100.58	17990.00	**7.68**	1447.00	14.03	282.00	-	-
ETTh1	96	2.52	**0.51**	1.88	633.00	2.88	369.00	2.87	2.00	**1.83**	26.00
	192	2.55	**0.50**	**2.00**	760.00	2.83	357.00	3.62	3.00	2.08	52.00
	336	2.53	**0.47**	**2.32**	825.00	2.90	341.00	4.90	4.00	3.20	86.00
	720	**3.87**	**0.56**	4.57	1188.00	4.40	301.00	7.87	6.00	5.63	165.00
ETTh2	96	2.95	**0.51**	**2.18**	633.00	2.83	372.00	3.23	2.00	2.23	45.00
	192	2.93	**0.48**	**2.52**	762.00	2.83	357.00	4.00	3.00	2.83	55.00
	336	2.95	**0.47**	**2.70**	827.00	2.90	340.00	5.70	4.00	5.18	268.00
	720	4.52	**0.67**	5.20	1189.00	**4.38**	304.00	8.85	7.00	5.60	189.00
ETTm1	96	12.02	**2.11**	**5.58**	1188.00	11.38	1534.00	12.98	8.00	9.78	334.00
	192	12.03	**2.12**	**5.73**	1342.00	11.42	1596.00	16.12	11.00	7.95	606.00
	336	13.72	**2.47**	**5.77**	1149.00	11.60	1502.00	22.33	17.00	16.92	394.00
	720	19.40	**3.47**	**5.70**	1140.00	18.15	1474.00	36.10	31.00	30.53	817.00
ETTm2	96	12.02	**2.07**	8.45	2656.00	11.35	1535.00	12.82	7.00	**7.65**	310.00
	192	12.00	**2.09**	9.55	3278.00	11.55	1580.00	16.07	11.00	**7.80**	561.00
	336	14.65	**2.74**	**10.37**	3642.00	11.62	1504.00	22.30	17.00	15.25	864.00
	720	19.87	**3.14**	20.92	5833.00	**18.12**	1479.00	36.18	31.00	27.57	1459.00
Weather	96	8.78	**1.58**	9.92	3442.00	8.48	1178.00	10.20	6.00	**6.20**	772.00
	192	8.97	**1.60**	15.15	4796.00	8.80	1192.00	12.80	18.00	**7.22**	1553.00
	336	11.17	**1.93**	22.17	6344.00	**8.83**	1157.00	17.63	15.00	10.68	2694.00
	720	**10.05**	**2.73**	44.33	11565.00	13.78	1142.00	28.18	29.00	-	-

The missing TACTiS-2 results are due to OOM, as its attention encoding and copula sampling require more GPU memory as the prediction horizon increases. As shown in Table 1, when averaged over four prediction lengths, TimeGDC achieves the lowest CRPS on all six datasets. It reduces CRPS by 24.9%, 15.6%, 13.2%, 17.3%, 10.0%, and 18.9% on Electricity, ETTh1, ETTh2, ETTm1, ETTm2, and Weather, respectively.

For point forecasting, TimeGDC also achieves the best average NMAE on all six datasets. It reduces NMAE by 5.5%, 11.0%, 9.1%, 1.5%, 5.8%, and 13.4%, respectively, with an average relative reduction of 7.7%. These results show that TimeGDC improves uncertainty estimation while maintaining strong point forecasting accuracy.

For joint distribution quality, TimeGDC achieves the lowest average ES on all six datasets, with an average relative reduction of 13.4%. These results support its effectiveness in modeling joint predictive distributions.

4.5 Computational Efficiency

To evaluate computational efficiency, all experiments are conducted on an NVIDIA GeForce RTX 4090 GPU with an input length of 96. Training time denotes the cumulative GPU-synchronized optimization time over all epochs. Inference time denotes the total GPU-synchronized time required by each method to generate 100 predictive trajectories for every test window over the complete test set. Data preprocessing, validation, checkpointing, and metric computation are excluded.

As shown in Table 2, TimeGDC achieves the lowest inference time in all 24 settings. Its single-pass mixture prediction avoids iterative denoising and costly dependency sampling, substantially reducing inference cost while maintaining reasonable training time.

4.6 Ablation Experiments

We assess the contributions of the Gaussian mixture (GMM), contamination-aware likelihood (CA), low-rank covariance, and temporal smoothing under the standard forecasting setting. The w/o GMM variant sets $K = 1$ while retaining CA and the smoothed low-rank covariance. The w/o DTSLC variant uses a diagonal covariance, removing both the low-rank covariance and temporal smoothing. The w/o Smooth variant retains the low-rank covariance but removes $\mathcal{L}_{\mathrm{TP}}$. The w/o CA variant replaces CA with standard NLL while retaining all other components. All other settings are fixed. CRPS, NMAE, and ES are averaged over the six datasets and prediction horizons $T \in \{96, 192, 336, 720\}$. Since CA is designed to reduce the influence of potentially contaminated observations, we further assess its robustness under controlled perturbations in Section 4.7.

Table 3: Ablation study of TimeGDC. CRPS, NMAE, and Energy Score (ES) are averaged over six datasets and prediction horizons $T \in \{96, 192, 336, 720\}$; lower values are better. Percentages in parentheses denote relative increases over TimeGDC, calculated from the displayed values. Best results are determined using unrounded averages and shown in bold.

Variant	GMM	CA	LR Cov.	Smooth	CRPS	NMAE	ES
TimeGDC	✓	✓	✓	✓	**0.345**	**0.356**	**6.069**
w/o CA	✓	–	✓	✓	0.346 (+0.29%)	0.356 (+0.00%)	6.075 (+0.10%)
w/o GMM	–	✓	✓	✓	0.349 (+1.16%)	0.361 (+1.40%)	6.132 (+1.04%)
w/o DTSLC	✓	✓	–	–	0.351 (+1.74%)	0.363 (+1.97%)	6.091 (+0.36%)
w/o Smooth	✓	✓	✓	–	0.349 (+1.16%)	0.358 (+0.56%)	6.138 (+1.14%)

Table 4: Robustness comparison with approximately 10% of target positions perturbed. Lower scores are better. Δ is the relative reduction (%) from w/o CA to w/ CA, calculated from the displayed three-decimal values; positive values indicate improvement.

Dataset	CRPS			NMAE			ES		
	w/o CA	w/ CA	Δ(%)	w/o CA	w/ CA	Δ(%)	w/o CA	w/ CA	Δ(%)
Electricity	0.286	**0.271**	5.24	0.282	**0.279**	1.06	5.455	**5.065**	7.15
ETTh1	0.477	**0.464**	2.73	0.521	**0.510**	2.11	9.136	**8.795**	3.73
ETTh2	0.353	**0.332**	5.95	0.330	**0.323**	2.12	5.931	**5.366**	9.53
ETTm1	0.437	**0.435**	0.46	0.460	0.460	0.00	8.265	**8.197**	0.82
ETTm2	0.296	**0.273**	7.77	0.277	**0.275**	0.72	5.154	**4.588**	10.98
Weather	0.407	**0.379**	6.88	0.294	**0.292**	0.68	6.035	**5.606**	7.11

As shown in Table 3, each component contributes to the overall performance of TimeGDC. GMM provides the most consistent gains across the three metrics: removing it degrades CRPS, NMAE, and ES by 1.16%, 1.40%, and 1.04%, respectively. DTSLC has the strongest effect on CRPS and NMAE, which increase by 1.74% and 1.97% without structured covariance modeling, indicating its importance for predictive distribution quality and point accuracy. Removing temporal smoothing also degrades all three metrics, with particularly clear increases in CRPS and ES. In contrast, removing CA causes only minor changes under the standard setting, including a 0.29% increase in CRPS. This is consistent with its primary role in robust learning, which is further demonstrated under controlled perturbations in Section 4.7. Additional ACF results are provided in Appendix D.

4.7 Robustness under Data Contamination

To isolate the contribution of the contamination-aware likelihood (CA), we evaluate robustness to contaminated training targets. After normalization using statistics fitted on clean training data, we add randomly signed spikes of magnitude 5 to approximately 10% of the window–channel target positions in the training set. Historical inputs and the validation and test sets remain unchanged; Table 4 reports results on the clean test set. We compare the contamination-aware likelihood (w/ CA) with standard negative log-likelihood (w/o CA). Both variants use the same perturbed training targets and otherwise identical experimental settings.

As shown in Table 4, w/ CA achieves lower CRPS and ES on all six datasets under the applied perturbations. Averaging the dataset-level relative reductions reported in the table, w/ CA reduces CRPS, NMAE, and ES by approximately 4.8%, 1.1%, and 6.6%, respectively. These gains suggest that the contamination-aware likelihood improves predictive robustness under this perturbation setting.

5 Conclusion

This paper proposes TimeGDC for efficient and robust long-term probabilistic time series forecasting. It models each channel's forecasting window with a Gaussian mixture to capture multimodal uncertainty, while dynamically smoothed low-rank covariances efficiently represent correlations among future steps. Its contamination-aware likelihood uses a fixed heavy-tailed reference distribution to reduce the influence of potentially contaminated observations during training. Experiments on six benchmarks show improved forecasting accuracy and predictive distribution quality, while the perturbation study provides evidence of robustness under the tested contamination setting.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language polishing and LaTeX formatting. All AI-assisted content was reviewed and verified by the authors, and all experiments were conducted and validated by the authors. We take full responsibility for the final content of this work, including its text, claims, figures, tables, and experimental results.

ETHICS STATEMENT

This work uses publicly available benchmark datasets and does not involve human subjects or personally identifiable information. The proposed method aims to improve probabilistic time series forecasting. In high-risk applications, its predictions should be interpreted with appropriate domain expertise and human oversight.

REPRODUCIBILITY STATEMENT

The model formulation and training objectives are described in Section 3, while the datasets, baseline methods, hyperparameters, sampling protocol, and evaluation settings are provided in Section 4. Definitions of the evaluation metrics, the selected numbers of Gaussian mixture components, and additional correlation analyses are included in the appendix.

REFERENCES

Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *arXiv preprint arXiv:2208.09399*, 2022.

Vaswani Ashish. Attention is all you need. *Advances in neural information processing systems*, 30: I, 2017.

Arjun Ashok, Étienne Marcotte, Valentina Zantedeschi, Nicolas Chapados, and Alexandre Drouin. Tactis-2: Better, faster, simpler attentional copulas for multivariate time series. In *International conference on learning representations*, volume 2024, pp. 22897–22924, 2024.

Shane Bergsma, Tim Zeyl, and Lei Guo. Sutranets: sub-series autoregressive networks for long-sequence, probabilistic forecasting. *Advances in Neural Information Processing Systems*, 36: 30518–30533, 2023.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. Springer, 2002.

Hang Ding, Xue Wang, Tian Zhou, and Tao Yao. Simdiff: Simpler yet better diffusion model for time series point forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 20781–20789, 2026.

Alexandre Drouin, Étienne Marcotte, and Nicolas Chapados. Tactis: Transformer-attentional copulas for time series. In *International Conference on Machine Learning*, pp. 5447–5493. PMLR, 2022.

Bradley Efron. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics: Methodology and distribution*, pp. 569–593. Springer, 1992.

Alireza Ermagun and David Levinson. Spatiotemporal traffic forecasting: review and proposed directions. *Transport Reviews*, 38(6):786–814, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Tilmann Gneiting and Matthias Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1(1):125–151, 2014.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Lu Han, Han-Jia Ye, and De-Chuan Zhan. The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):7129–7142, 2024.

Genshiro Kitagawa and Will Gersch. *Smoothness priors analysis of time series*, volume 116. Springer Science & Business Media, 1996.

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.

Vincent Le Guen and Nicolas Thome. Probabilistic time series forecasting with shape and temporal diversity. *Advances in neural information processing systems*, 33:4427–4440, 2020.

Yuxin Li, Wenchao Chen, Xinyue Hu, Bo Chen, Mingyuan Zhou, et al. Transformer-modulated diffusion models for probabilistic multivariate time series forecasting. In *International Conference on Learning Representations*, volume 2024, pp. 18604–18622, 2024.

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, 2024.

James E Matheson and Robert L Winkler. Scoring rules for continuous probability distributions. *Management science*, 22(10):1087–1096, 1976.

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*.

Karl Pearson. X. contributions to the mathematical theory of evolution.—ii. skew variation in homogeneous material. *Philosophical Transactions of the Royal Society of London, Series A: Containing Papers of a Mathematical or Physical Character*, (186):343–414, 1895.

Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31, 2018.

Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs Bergmann, and Roland Vollgraf. Multivariate probabilistic time series forecasting via conditioned normalizing flows. *arXiv preprint arXiv:2002.06103*, 2020.

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.

David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. *Advances in neural information processing systems*, 32, 2019.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

David Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pp. 1177–1178, 2010.

Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, 90:106181, 2020.

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.

Yanru Sun, Zongxia Xie, Haocheng Wang, Xin Huang, and Qinghua Hu. Solar wind speed prediction via graph attention network. *Space Weather*, 20(7):e2022SW003128, 2022.

Binh Tang and David S Matteson. Probabilistic transformer for time series analysis. *Advances in neural information processing systems*, 34:23592–23608, 2021.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in neural information processing systems*, 34:24804–24816, 2021.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

Xingjian Wu, Xiangfei Qiu, Hongfan Gao, Jilin Hu, Bin Yang, and Chenjuan Guo. k^2 vae: A koopman-kalman enhanced variational autoencoder for probabilistic time series forecasting. *arXiv preprint arXiv:2505.23017*, 2025.

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

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.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pp. 27268–27286. PMLR, 2022.

Table 5: Overview of dataset information and data splits.

Dataset	Variables	Sampling Frequency	Train	Val./Test
Electricity	321	1 hour	70%	10%/20%
ETTh1	7	1 hour	60%	20%/20%
ETTh2	7	1 hour	70%	10%/20%
ETTm1	7	15 minutes	70%	10%/20%
ETTm2	7	15 minutes	70%	10%/20%
Weather	21	10 minutes	70%	10%/20%

A Dataset Details

Table 5 summarizes the dataset statistics and train/validation/test splits used in our experiments.

B Definitions of Evaluation Metrics

For each forecasting instance and target channel, let $\mathbf{y} \in \mathbb{R}^T$ denote the observed forecasting window and let $\{\hat{\mathbf{y}}^{(s)}\}_{s=1}^{S}$ denote S samples from the predictive distribution. CRPS is globally normalized over all test windows, target channels, and forecasting steps, while ES is computed for each forecasting window and channel and then averaged.

Continuous Ranked Probability Score. Following the CSDI evaluation protocol, we approximate CRPS using the quantile set $\mathcal{Q} = \{0.05, 0.10, \ldots, 0.95\}$. Let $\hat{y}_{\tau,n,c,t}$ denote the empirical τ-quantile for test window n, target channel c, and forecasting step t. The quantile loss is defined as

$$\rho_\tau(u) = u\left(\tau - \mathbb{I}\{u < 0\}\right). \tag{17}$$

The globally normalized CRPS approximation is

$$\widehat{\mathrm{CRPS}} = \frac{2}{|\mathcal{Q}|} \sum_{\tau \in \mathcal{Q}} \frac{\sum_{n=1}^{N} \sum_{c=1}^{C_y} \sum_{t=1}^{T} \rho_\tau\left(y_{n,c,t} - \hat{y}_{\tau,n,c,t}\right)}{\sum_{n=1}^{N} \sum_{c=1}^{C_y} \sum_{t=1}^{T} |y_{n,c,t}|}. \tag{18}$$

Energy Score. The ES evaluates the overall quality of the joint predictive distribution. Its Monte Carlo estimate is

$$\widehat{\mathrm{ES}} = \frac{1}{S} \sum_{s=1}^{S} \left\| \hat{\mathbf{y}}^{(s)} - \mathbf{y} \right\|_2 - \frac{1}{2S^2} \sum_{s=1}^{S} \sum_{s'=1}^{S} \left\| \hat{\mathbf{y}}^{(s)} - \hat{\mathbf{y}}^{(s')} \right\|_2. \tag{19}$$

The first term measures the distance between predictive samples and the observation, while the second term accounts for the dispersion of the predictive samples.

Normalized Mean Absolute Error. Since the forecasting targets are standardized using statistics from the training set, NMAE is computed as the mean absolute error in the standardized space:

$$\mathrm{NMAE} = \frac{1}{T} \sum_{t=1}^{T} |\hat{y}_t - y_t|, \tag{20}$$

where $\hat{\mathbf{y}}$ denotes the point prediction.

C Gaussian Mixture Component Settings

Table 6 reports the selected number of Gaussian mixture components for each dataset and prediction length. Electricity uses $K = 3$, while the other five datasets use $K = 2$. The selected values do not vary with the prediction length.

Table 6: Number of Gaussian mixture components K used for each dataset and prediction length.

Dataset	96	192	336	720
Electricity	3	3	3	3
ETTh1	2	2	2	2
ETTh2	2	2	2	2
ETTm1	2	2	2	2
ETTm2	2	2	2	2
Weather	2	2	2	2

D CORRELATION ANALYSIS

To quantitatively evaluate the capability of capturing temporal correlations, we use the autocorrelation function (ACF). ACF measures the correlation between observations of a time series at different time lags, and is commonly used to characterize the temporal dependency structure within a sequence Brockwell & Davis (2002). Specifically, for a time series $\mathbf{x} = \{x_t\}_{t=1}^{T}$, the ACF at lag l can be defined as follows:

$$\rho(l) = \frac{\sum_{t=1}^{T-l}(x_t - \bar{x})(x_{t+l} - \bar{x})}{\sum_{t=1}^{T}(x_t - \bar{x})^2}, \tag{21}$$

where $\bar{x}$ denotes the mean of the series. A larger absolute value of $\rho(l)$ indicates a stronger dependency between observations separated by l time steps.

Based on ACF, we further adopt ACF-MSE as the evaluation metric to measure the difference between the temporal correlation patterns of the predicted series and the ground-truth series:

$$\text{ACF-MSE} = \frac{1}{L}\sum_{l=1}^{L}\left(\rho_{\text{pred}}(l) - \rho_{\text{true}}(l)\right)^2, \tag{22}$$

where L is the maximum lag considered. In our experiments, we set L to half of the prediction length, which allows ACF-MSE to evaluate temporal dependencies effectively while avoiding overly long lags that may lead to unstable autocorrelation estimates. A lower ACF-MSE indicates that the predicted series better preserves the temporal correlation structure of the ground truth.

To provide a more intuitive comparison, we visualize the ACF curves of the ground-truth series, TimeGDC predictions, and the variant without DTSLC, as shown in Fig. 3. Due to space limitations, we select representative datasets and prediction horizons for visualization. The closer the predicted ACF curve is to the ground-truth ACF curve, the better the model preserves the temporal correlation structure. As shown in Fig. 3, TimeGDC produces ACF curves that are closer to the ground truth than NoDTSLC in the selected cases, suggesting that the proposed DTSLC module helps the model better capture temporal correlation patterns.

Beyond the qualitative visualization, we further conduct a comprehensive quantitative comparison using ACF-MSE over all datasets and prediction horizons. The comparison between TimeGDC and NoDTSLC is summarized in Table 7.

As shown in Table 7, TimeGDC achieves lower average ACF-MSE than NoDTSLC on all six datasets, demonstrating its stronger ability to preserve temporal correlation structures. These results indicate that dynamic temporal smoothing low-rank covariance modeling can effectively enhance the characterization of temporal correlations across different time steps. Although NoDTSLC obtains slightly lower ACF-MSE in a few individual settings, TimeGDC consistently achieves better average performance across datasets. This suggests that DTSLC provides more stable temporal correlation modeling overall, rather than only improving isolated forecasting horizons.

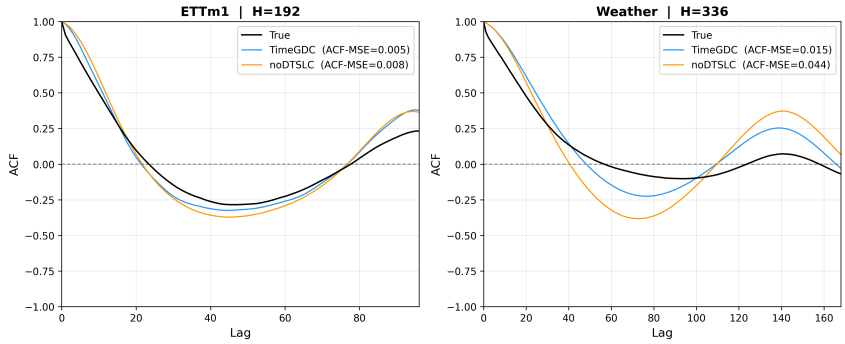

Figure 3: ACF visualization on ETTm1 with prediction length 192 and Weather with prediction length 336. Compared with the variant without DTSLC, TimeGDC produces ACF curves that are closer to the ground-truth curves, demonstrating the effectiveness of DTSLC in preserving temporal correlation structures.

Table 7: ACF-MSE comparison between TimeGDC and its variant without DTSLC. The best results are highlighted in red. Averages and rankings are calculated from unrounded values.

Dataset	Pred Len	TimeGDC	NoDTSLC
Electricity	96	0.003	0.004
	192	0.004	0.005
	336	0.004	0.006
	720	0.005	0.007
	avg	0.004	0.005
ETTh1	96	0.021	0.027
	192	0.021	0.035
	336	0.025	0.027
	720	0.019	0.030
	avg	0.022	0.029
ETTh2	96	0.058	0.102
	192	0.068	0.103
	336	0.070	0.091
	720	0.068	0.051
	avg	0.066	0.087
ETTm1	96	0.005	0.006
	192	0.005	0.008
	336	0.007	0.012
	720	0.013	0.023
	avg	0.007	0.012
ETTm2	96	0.025	0.045
	192	0.034	0.041
	336	0.046	0.039
	720	0.037	0.042
	avg	0.036	0.042
Weather	96	0.008	0.010
	192	0.015	0.021
	336	0.015	0.044
	720	0.033	0.027
	avg	0.018	0.026