arXiv:2608.08010v1 [cs.LG] 8 Aug 2026

Ground-Truth Neighborhood Regularization for Reinforcement Learning Post-Training of Time Series Foundation Models

Jianqi Zhang[1], **Xingyu Zhang**[1], **Zeen Song**[1], **Changwen Zheng**[1], **Fanjiang Xu**[1], **Wenwen Qiang**[1]

[1]Institute of Software, Chinese Academy of Sciences, Beijing, China
jluzhangjianqi@163.com, xingyuzhang@mails.ucas.ac.cn, zeensong@qq.com,
changwen@iscas.ac.cn, fanjiang@iscas.ac.cn, qiangwenwen@iscas.ac.cn

Preprint. Under review.

Abstract

Time series forecasting (TSF) plays an important role in a wide range of real-world applications. Recently, time series foundation models (TSFMs), pretrained on large-scale datasets, have demonstrated strong generalization capabilities and emerged as an important paradigm for TSF. Reinforcement learning (RL) post-training has consequently attracted growing attention as a means of further improving their performance on downstream tasks. However, we find that, in certain forecast regions, RL post-training may gradually shift the output distributions of TSFMs away from the ground truth, thereby limiting their performance. We refer to this phenomenon as **suboptimal collapse**. Our analysis suggests that difficulty in initially sampling high-quality trajectories near the ground truth is an important contributing factor to suboptimal collapse. To address this issue, we propose Ground-Truth Neighborhood Regularization (GTN-R) for RL post-training of TSFMs. GTN-R uses the ground truth as a reference for locating high-quality regions and guides the model's probability mass toward the ground-truth neighborhood. This increases the probability of sampling high-quality trajectories, mitigates suboptimal collapse, and improves performance. Moreover, GTN-R can be flexibly integrated into various RL methods for TSFMs. Extensive experiments show its effectiveness.

Introduction

Time series forecasting (TSF) is a fundamental task in a wide range of applications, including energy scheduling (Boussif et al. 2024; Novo et al. 2022; Lara-Benítez et al. 2020; Deb et al. 2017), traffic management (Fang et al. 2023; Li et al. 2022; Wang et al. 2022; Shekhar and Williams 2007), and weather forecasting (Abhishek et al. 2012; Karevan and Suykens 2020; Campbell and Diebold 2005; Hamilton, Brown, and Keim 2007). Recently, time series foundation models (TSFMs), pretrained on large-scale time series data, have demonstrated strong generalization across diverse scenarios (Woo et al. 2024). However, the general capabilities acquired during pretraining may not fully adapt to specific downstream tasks, making post-training necessary to improve performance in target scenarios further (Qiao et al. 2025). Recent studies (Qi et al. 2025; Li et al. 2026) have shown that, compared with supervised fine-tuning (SFT), reinforcement learning (RL) post-training can more effectively unlock the forecasting capabilities of TSFMs on downstream tasks, thereby attracting increasing attention.

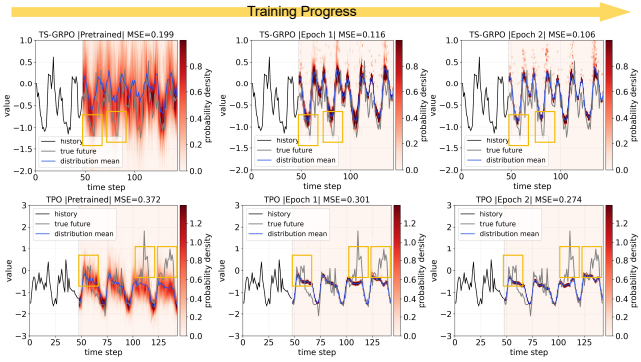

Figure 1: Illustration of suboptimal collapse during RL post-training of Moirai-small. Each row shows the evolving predictive distribution for a training sample, with the RL method, processed epoch, and MSE reported above each panel. The TSFM used is Moirai-small.As training progresses, distribution within the yellow boxes shifts toward regions far from the ground truth, indicating suboptimal collapse. More results are in Appendix "More Visualization."

Although RL post-training improves the forecasting performance of TSFMs, we identify a notable issue during training. Specifically, ideally, as training progresses, the model's predictive distribution should gradually concentrate around the ground truth. However, we observe that, in many cases, although the overall MSE decreases, the predictive distribution in some regions (yellow boxes) gradually concentrates outside the ground-truth neighborhood, as shown in Fig.1. We term this phenomenon **suboptimal collapse**. It may trap subsequent optimization in low-quality regions far from the ground truth and limit forecasting performance. From an optimization perspective, suboptimal collapse may arise because the model initially struggles to sample high-quality trajectories near the ground truth in certain regions. In this case, the relatively better trajectories within a sampled group are often located in low-quality regions far from the ground truth. Since RL optimizes the model by increasing the probabilities of relatively better trajectories (Qi et al. 2025; Li et al. 2026), updates in the above sampled group further

widen the probability gap between the low-quality region and the ground-truth neighborhood. This makes the model more likely to sample trajectories from the low-quality region and, consequently, makes the relatively better trajectories in subsequent sampled groups more likely to be in the low-quality region. Repeated occurrences of this process form a detrimental self-reinforcing cycle, which may eventually drive the probability of the ground-truth neighborhood toward zero, thereby leading to suboptimal collapse. In Sections "Theoretical Analysis" and "Empirical Analysis", we further validate this intuitive conjecture from theoretical and empirical perspectives, respectively.

An intuitive solution is to directly include the ground-truth trajectory in the sampled group. However, because the ground-truth participates in the updates as a fixed sample with the highest reward, the model may instead learn to imitate it in a supervised manner, causing RL post-training to gradually degenerate into a process resembling behavior cloning (Yan et al. 2026). This may weaken the model's exploration capability and further impair its generalization ability (Wang et al. 2025; Weltevrede et al. 2024; Chu et al. 2025). Our experiments in Section "Empirical Analysis" further support this argument: although this approach mitigates suboptimal collapse on the training data, it undermines the model's exploration capability and generalization performance. Therefore, although the ground-truth trajectory can serve as a reference for high-quality trajectories, it should not participate in the updates as a fixed high-reward sample.

Based on the above analysis, we propose Ground-Truth Neighborhood Regularization (GTN-R), a regularization method for RL post-training of TSFMs. Instead of directly including the ground truth in the updates as a fixed high-reward sample, GTN-R uses it only as a reference for high-quality trajectories, guiding the model to increase the probability of sampling high-quality trajectories. Consequently, GTN-R mitigates suboptimal collapse while preserving the model's autonomous exploration capability, thereby further improving performance. Specifically, GTN-R introduces two distributional constraints into the original RL objective: the neighborhood probability mass constraint and the within-neighborhood uniformity constraint. The former maximizes the probability mass assigned by the predictive distribution to the ground-truth neighborhood, thereby increasing the probability of sampling high-quality trajectories and mitigating suboptimal collapse. The latter encourages the distribution within this neighborhood to approach a uniform distribution, preventing probability mass from concentrating at only a few locations and thus preserving the model's exploration ability. We integrate GTN-R into multiple RL methods applicable to TSFMs. Experiments across multiple TSFMs and datasets show that GTN-R effectively mitigates suboptimal collapse and improves performance.

Our contributions are as follows: **1)** We identify **suboptimal collapse** in RL post-training of TSFMs and, through theoretical analysis and empirical studies, investigate its mechanism and negative impact on forecasting performance. **2)** To address this issue, we propose GTN-R, which mitigates suboptimal collapse by increasing the probability mass assigned to the ground-truth neighborhood while encouraging a near-uniform distribution within it, thereby improving forecasting performance. **3)** We validate the effectiveness of GTN-R across multiple RL methods for TSFMs.

Related Work

Recently, Time Series Foundation Models (TSFMs) have advanced rapidly (Miller et al. 2024; Kottapalli et al. 2025; Ye et al. 2026; Liu et al. 2025). Some TSFMs, such as Moment (Goswami et al. 2024), UniTS (Gao et al. 2024), TimesFM (Das et al. 2024), and Timer (Liu et al. 2024b), produce deterministic forecasts. However, stochastic disturbances, observation noise, and environmental changes make future time series inherently uncertain (Clark and Bjørnstad 2004; Yoon et al. 2022). Deterministic predictions cannot capture multiple possible futures or quantify predictive confidence, thereby limiting their ability to support uncertainty-aware decision-making in real-world (Liu et al. 2024a). This has motivated probabilistic TSFMs (Moirai (Woo et al. 2024), Moirai-MoE (Liu et al. 2024a), and Toto (Cohen et al. 2024)), which produced the distribution of future series.

For downstream-task post-training, the research community initially relies on supervised fine-tuning (SFT) to optimize TSFMs (Qiao et al. 2025). However, (Qiao et al. 2025; Qi et al. 2025; Li et al. 2026) shows that naive SFT cannot fully unlock TSFMs' forecasting potential. Recent studies (Qi et al. 2025; Li et al. 2026), therefore, apply RL methods, such as RLHF (Ouyang et al. 2022) and GRPO (Shao et al. 2024), to TSFM post-training and achieve better performance than SFT. However, we find that directly applying RL may lead to suboptimal collapse, as shown in Fig.1. This paper aims to mitigate suboptimal collapse in RL post-training of TSFMs and further improve forecasting performance.

Preliminaries

Time Series Forecasting (TSF).

In TSF, given a historical series $X \in \mathbb{R}^{T_h \times N}$, where T_h is the input length, N is the number of variables, the task is to predict the series of the next T_f time steps $Y \in \mathbb{R}^{T_f \times N}$.

Probabilistic Forecasting and Sampling in TSFMs

In this paper, we focus on probabilistic TSFMs because they explicitly model predictive distributions of future series, from which RL can conveniently sample trajectories for update. Probabilistic TSFMs mainly adopt two forecasting paradigms: one-shot forecasting and token-by-token forecasting (Liu et al. 2026). We next describe how these two get future predictive distributions and sample trajectories.

We first introduce the one-shot forecasting paradigm, which is adopted by Moirai (Woo et al. 2024). Given a historical series X, the model simultaneously predicts the distributions of all future time points through a single forward pass, yielding the set of all predictive distributions:

$$f_\theta(X) = \{P(\hat{y}_1 \mid X), ..., P(\hat{y}_{T_f} \mid X)\} = \mathcal{P}, \qquad (1)$$

where f_θ denotes the forecasting function and $P(\hat{y}_t \mid X)$ the predictive distribution of the t-th point of $\hat{Y}$. We then sample

from the point-wise distributions in $\mathcal{P}$ to construct multiple trajectories for training:

$$\{\hat{Y}^i\}_{i=1}^n \sim \mathcal{P}. \tag{2}$$

We next introduce the token-by-token forecasting paradigm, which is adopted by Moirai-MoE (Liu et al. 2024a) and Toto (Cohen et al. 2024). Given a historical series X, the future series Y is divided into contiguous segments of equal length $\{Y_{s1}, Y_{s2}, \ldots\}$, each corresponding to one token. The model first predicts the distribution of each point in Y_{s1}:

$$f_\theta(X) = \{P(\hat{y}_{s1,1} \mid X), ..., P(\hat{y}_{s1,T_p} \mid X)\} = \mathcal{P}_{s1}, \tag{3}$$

where T_p is the length of each patch, $P(\hat{y}_{s1,t} \mid X)$ is the t-th-point predictive distribution of $\hat{Y}_{s1}$. We then sample from the point-wise distributions in $\mathcal{P}_{s1}$ to construct multiple segment-level trajectories:

$$\{\hat{Y}_{s1}^i\}_{i=1}^n \sim \mathcal{P}_{s1}. \tag{4}$$

Next, for each $\hat{Y}_{s1}^i$, we concatenate it with the existing series to form a new series $(X, \hat{Y}_{s1}^i)$. Then it is fed into the model to predict the distribution of each point of the next segment, $\mathcal{P}_{s2}^i$=$\{P(\hat{y}_{s2,t} \mid X, \hat{Y}_{s1}^i)\}_{t=1}^{T_p}$. Next, we sample only one segment-level trajectory from $\mathcal{P}_{s2}^i$ as the continuation of $\hat{Y}_{s1}^i$. This process is repeated until each trajectory reaches the required prediction length.

Theoretical Analysis

In this section, we theoretically analyze a possible mechanism that may lead to suboptimal collapse. For a given future time point t and its predictive context $\mathcal{C}_t$, let $\mathcal{G} = \mathcal{N}_r(y_t)$ be the ground-truth neighborhood, and let $\mathcal{S}$ be a low-quality region such that $\mathcal{G} \cap \mathcal{S} = \varnothing$. Their union is not required to cover the entire point-wise prediction space. Let $p_{\theta_k,t}(\cdot \mid \mathcal{C}_t)$ be the model's predictive probability at iteration k. We define:

$$p_k = p_{\theta_k,t}(\mathcal{G} \mid \mathcal{C}_t), \qquad s_k = p_{\theta_k,t}(\mathcal{S} \mid \mathcal{C}_t). \tag{5}$$

Suppose that M trajectories are independently sampled at each iteration, and let $\hat{y}_t^i$ denote the prediction at time point t in the i-th trajectory. Let E_k denote the event that at least one sampled prediction lies in $\mathcal{G}$, i.e., $\hat{y}_t^i \in \mathcal{G}$ for some i, and let E_k^c denote its complement. We further let A_k denote the event that E_k^c occurs and the predictions at time point t in the positively reinforced trajectories (i.e., the better trajectories among the M samples) lie in $\mathcal{S}$.

Our analysis is based on the following conditions.

Initial sampling difficulty. The probability p_k assigned to the ground-truth neighborhood at time point t is low.

Low-quality-dominance update. When no sampled prediction at time point t lies in $\mathcal{G}$, the values of the points of better trajectories that dominate the update are likely to lie in a certain low-quality region $\mathcal{S}$. Formally, there exists a sufficiently large $\rho \in (0, 1)$ such that

$$p(A_k \mid \mathcal{F}_k, E_k^c) \geq \rho, \tag{6}$$

where $\mathcal{F}_k$ denotes the training history before iteration k.

Relative amplification. Since RL updates increase the probabilities of relatively better sampled trajectories (Li et al. 2026), when A_k occurs, the update increases the probability of $\mathcal{S}$ more than that of $\mathcal{G}$:

$$\frac{s_{k+1}}{s_k} \geq (1+\eta)\frac{p_{k+1}}{p_k}, \eta > 0. \tag{7}$$

We emphasize that these conditions characterize a possible failure mode and need not hold for every update of a specific RL method. Then, we can derive:

Proposition 1. *Under the above conditions, whenever A_k occurs, after the update, the lower bound on the probability of the next low-quality-dominated update at time point t increases:*

$$p(A_{k+1} \mid \mathcal{F}_{k+1}) \geq \rho(1 - h_{k+1})^M, \tag{8}$$

where $h_k = \frac{p_k}{p_k + s_k}$. If such updates occur for L consecutive iterations, then

$$p_{k+L} \leq \frac{1}{1 + (1+\eta)^L s_k/p_k} \longrightarrow 0 \qquad \textit{as } L \to \infty. \tag{9}$$

Proposition 1 characterizes a sufficient self-reinforcing mechanism for suboptimal collapse at a forecasting point. Intuitively, when the initial probability assigned to its ground-truth neighborhood is low (i.e., p_0 is low), a low-quality-dominated update (A_0) is more likely to occur. Once such an update occurs, Eq.8 shows that the lower bound on the probability of the next low-quality-dominated update increases, making such updates more likely to persist. If this process persists, Eq.9 implies that the ground-truth-neighborhood probability at this forecasting point can approach zero, leading to suboptimal collapse.

Empirical Analysis

In this section, we provide empirical evidence consistent with the self-reinforcing process characterized in Proposition 1 and investigate the effect of adding the ground-truth trajectory to the sampled group during training.

In the first experiment, we empirically analyze suboptimal collapse. Specifically, we select three TSFMs as backbones—Moirai-small (Woo et al. 2024), Toto (Cohen et al. 2024), and Moirai-MoE$_s$ (Liu et al. 2024a). And we apply TPO (Qi et al. 2025), TimeRFT (Li et al. 2026), and TS-GRPO, respectively, for RL post-training. TS-GRPO is a simple GRPO-based post-training method applicable to different TSFMs, with implementation details provided in Appendix "TS-GRPO". We first perform a complete RL post-training run. We then randomly select 1,000 training samples and identify the time points at which collapse does and does not occur in these samples. After training, a time point is considered collapsed if its ground-truth-neighborhood probability is below 0.001. The neighborhood radius is set to 1 for Loop Seattle and 0.1 for all other datasets. We then retrain the models and record, throughout training, the mean log probability assigned by the model's output distribution to the ground-truth neighborhoods of these two groups of points. The results are shown in Fig.2a. We observe that: 1) regions of collapse typically have lower initial ground-truth-neighborhood probability, which further decreases during training; and 2) regions of non-collapse have higher initial

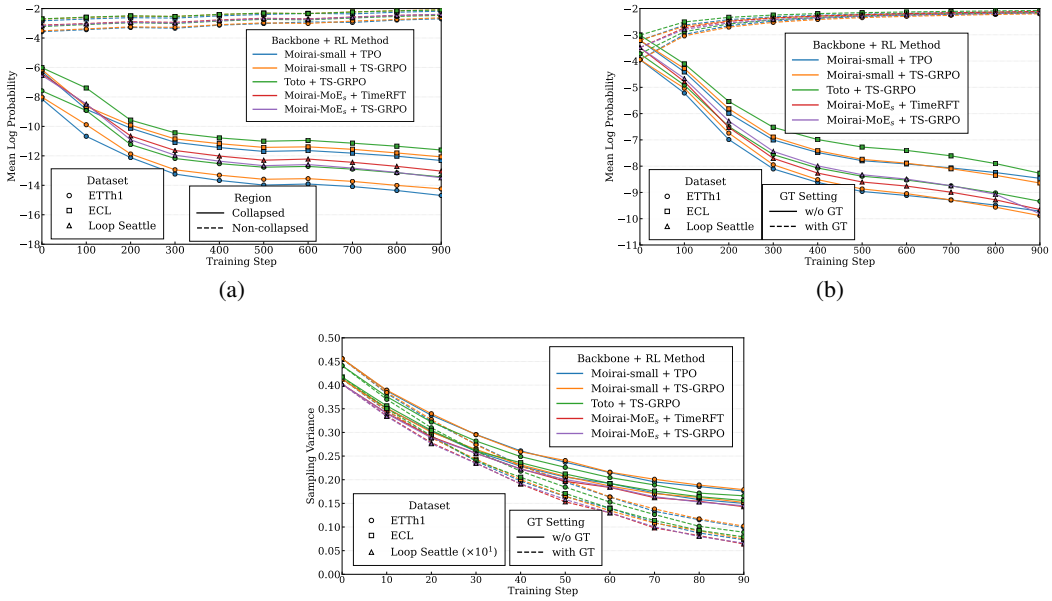

Figure 2: (a) Mean log probability assigned to the ground-truth neighborhoods of collapsed and non-collapsed points during post-training. Mean ground-truth-neighborhood log probability (b) and sampling variance (c) with and without including the ground-truth trajectory. Train step denotes the number of processed batches. In (c), $\times 10^1$ indicates that the actual variances of the dataset are 10 times the plotted values.

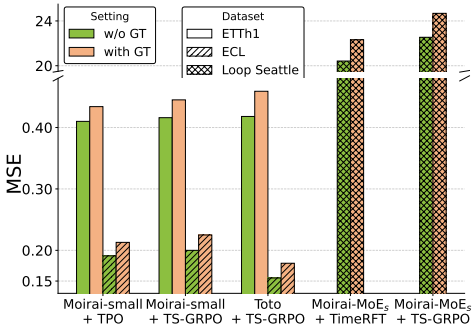

Figure 3: Test-set forecasting performance with and without including the ground-truth trajectory in the sampled group.

probability, which continues to increase throughout training. These results are consistent with the conditional self-reinforcing process characterized in Proposition 1: when the model initially struggles to sample high-quality trajectories in certain regions, a detrimental self-reinforcing cycle can emerge, eventually leading to suboptimal collapse.

In the second experiment, we examine the effect of adding the ground-truth trajectory to the sampled group during training. Specifically, the TSFMs, RL methods, and observed train data selection settings are the same as those described above. We train the models under two settings: with and without the ground-truth trajectory included. We clip the ground-truth reward at 1.5 times the maximum sampled reward within its group to prevent an excessively high ground-truth reward from compromising training stability. During training, we record three metrics: 1) the mean log probability assigned by the model's output distribution to the ground-truth neighborhoods of these selected samples; 2) the sampling variance of the predictive trajectories for these selected samples; and 3) the forecasting performance of the trained model on the test set. The results are in Fig.2b, 3 and 2c. We observe that directly adding the ground-truth trajectory substantially increases the probability of the ground-truth neighborhood, thereby mitigating suboptimal collapse. However, both the sampling variance and test-set performance decrease. These results indicate that the ground-truth trajectory can serve as a reference for high-quality trajectories to mitigate suboptimal collapse, but should not directly participate in policy updates as a fixed high-reward sample.

Method

In this section, we propose GTN-R, a regularization method for RL post-training of TSFMs. GTN-R uses the ground truth only to identify high-quality regions and increase the probability of sampling trajectories from them, without directly incorporating ground-truth trajectories into policy updates. This mitigates suboptimal collapse while preserving exploration, thereby improving performance. Specifically, we first introduce the overall pipeline, then describe the regularization constraints, and finally present the learning objective.

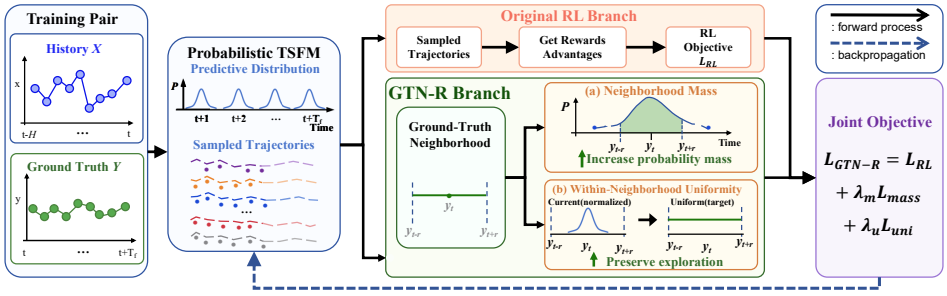

Figure 4: Overall framework of our method.

Overall Pipeline

In this subsection, we introduce the overall pipeline of GTN-R. Fig.4 illustrates the overall training framework of GTN-R. Specifically, we adopt a TSFM capable of producing predictive distributions as the backbone model. Given a train date, we compute the Ground-Truth Neighborhood Regularization term, as detailed in the next subsection, and incorporate it into the original training objective. Finally, we update the model using the resulting objective under the RL paradigm.

Ground-Truth Neighborhood Regularization

In this subsection, we describe how to compute the proposed regularization terms based on the ground-truth neighborhood. Specifically, we first formally define the ground-truth neighborhood. We then introduce how to get the first regularization term: the neighborhood probability mass constraint. Finally, we introduce how to get the second regularization term: the within-neighborhood uniformity constraint.

Firstly, we formally define the ground-truth neighborhood. Given a historical series X and the corresponding ground-truth future series:

$$Y = \{y_1, y_2, ..., y_{T_f}\}. \tag{10}$$

We construct a neighborhood centered at each ground-truth value y_t:

$$\mathcal{N}_r(y_t) = [y_t - r,\ y_t + r], \tag{11}$$

where r is a hyperparameter. $\{\mathcal{N}_r(y_t)\}_{t=1}^{T_f}$ constitute the ground-truth neighborhood $\mathcal{N}_r(Y)$.

Secondly, we introduce how to compute the first regularization term, namely the neighborhood probability mass constraint. Specifically, let $p_{\theta,t}(z \mid \mathcal{C}_t^i)$ denote the probability density at z of the predictive distribution at the t-th future time point of the i-th sampled trajectory. For one-shot forecasting, $\mathcal{C}_t^i = X$; for token-by-token forecasting, $\mathcal{C}_t^i$ additionally includes the previously generated segments. Next, we compute the probability mass assigned to the neighborhood $\mathcal{N}_r(y_t)$ under the context of the i-th trajectory:

$$p^i(\mathcal{N}_r(y_t)) = \int_{y_t - r}^{y_t + r} p_{\theta,t}(z \mid \mathcal{C}_t^i)\, dz. \tag{12}$$

The integral in Eq.12 can be computed using Gauss–Legendre quadrature (Swarztrauber 2003). Based on this, we define the neighborhood probability mass constraint for the i-th sampled trajectory as the mean log-probability of $p^i(\mathcal{N}_r(y_t))$ over all future time points:

$$\mathcal{L}_{\text{mass}}^i = \frac{1}{T_f} \sum_{t=1}^{T_f} \log\big(p^i(\mathcal{N}_r(y_t))\big) \tag{13}$$

The reason for using the logarithm is that it imposes a stronger penalty on forecast points with extremely low probability, preventing them from being overlooked in the constraint. Our ablation study validates the rationality of this design.

For token-by-token forecasting, different sampled trajectories have different predictive distributions. We thus average the trajectory-wise constraints to get the final neighborhood probability mass constraint: $\mathcal{L}_{\text{mass}} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_{\text{mass}}^i$, where n is the number of trajectories sampled for each input during one update of the RL algorithm. For one-shot forecasting, the predictive distributions depend only on X and are identical across sampled trajectories. That makes $\mathcal{L}_{\text{mass}}^1 = \cdots = \mathcal{L}_{\text{mass}}^n$. Therefore, we simply set $\mathcal{L}_{\text{mass}} = \mathcal{L}_{\text{mass}}^i$ in this case. By maximizing $\mathcal{L}_{\text{mass}}$, we encourage the model to assign more probability mass to the ground-truth neighborhoods under different trajectory contexts, thereby increasing the likelihood of sampling high-quality trajectories near the ground truth.

Finally, we describe how to compute the second regularization term, namely, the within-neighborhood uniformity constraint. Specifically, we first normalize the predictive density of the i-th sampled trajectory in $\mathcal{N}_r(y_t)$:

$$\widetilde{p}_{\theta,t}^i(z \mid \mathcal{C}_t^i) = p_{\theta,t}(z \mid \mathcal{C}_t^i)/p^i(\mathcal{N}_r(y_t)), \quad z \in \mathcal{N}_r(y_t). \tag{14}$$

Meanwhile, we define the uniform distribution in $\mathcal{N}_r(y_t)$ as:

$$u_t(z) = 1/2r, \quad z \in \mathcal{N}_r(y_t). \tag{15}$$

Let $\widetilde{P}_{\theta,t}^i$ and U_t denote the distributions corresponding to $\widetilde{p}_{\theta,t}^i(z \mid \mathcal{C}_t)$ and $u_t(z)$, respectively. Based on this, we define the within-neighborhood uniformity constraint for the i-th sampled trajectory as the negative mean KL divergence between these two distributions over all future time points:

$$\mathcal{L}_{\text{uni}}^i = -\frac{1}{T_f} \sum_{t=1}^{T_f} D_{\text{KL}}\left(\widetilde{P}_{\theta,t}^i \,\|\, U_t\right). \tag{16}$$

The D_{KL} also can be computed using Gauss–Legendre quadrature (Swarztrauber 2003). Similar to $\mathcal{L}_{\text{mass}}$, for token-by-token forecasting, $\mathcal{L}_{\text{uni}}$ is defined as the mean of $\mathcal{L}_{\text{uni}}^i$ over

all sampled trajectories: $\mathcal{L}_{\text{uni}} = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_{\text{uni}}^{i}$. For one-shot forecasting, $\mathcal{L}_{\text{uni}} = \mathcal{L}_{\text{uni}}^{i}$. By maximizing $\mathcal{L}_{\text{uni}}$, we encourage a uniform distribution within $\mathcal{N}_r(Y)$, preventing the probability mass from concentrating on only a few locations and thereby preserving the model's exploration capability.

Learning Objective of GTN-R

After obtaining the neighborhood probability mass constraint and the within-neighborhood uniformity constraint, we incorporate both into the original RL objective, yielding the GTN-R learning objective to be maximized:

$$\mathcal{L}_{GTN-R} = \mathcal{L}_{RL} + \lambda_m \mathcal{L}_{mass} + \lambda_u \mathcal{L}_{uni}, \qquad (17)$$

where, $\mathcal{L}_{\text{RL}}$ denotes the original RL objective, while λ_m and λ_u are the weights of the two regularization terms. Maximizing this objective enables the model to mitigate suboptimal collapse while preserving its autonomous exploration capability, thereby further improving forecasting performance.

Experiments

In this section, we empirically evaluate GTN-R. We first introduce the experimental setup, then present comparative results of multiple datasets and visualize how GTN-R affects the model's output distributions during training. Additional visualizations, ablations, computational-cost, and hyperparameter-sensitivity are provided in the Appendix.

Experimental Settings

Datasets:

For general time series forecasting, we use eight real-world datasets: ETT (ETTh1, ETTh2, ETTm1, and ETTm2), ECL, Weather, Loop Seattle, and ENTSO-e Load. For zero-shot forecasting, we use the ETT and Weather datasets. The data splits for Loop Seattle and ENTSO-e Load follow TimeRFT (Li et al. 2026), while those for the remaining datasets follow (Qiao et al. 2025).

Baselines:

We mainly follow the post-training evaluation framework of (Qiao et al. 2025) and adopt Moirai (Woo et al. 2024), Toto (Cohen et al. 2024), Moirai-MoE (Liu et al. 2024a), and UniTS (Gao et al. 2024) as the time series foundation models. Following (Qiao et al. 2025), we use the x128 pretrained checkpoint for UniTS. Since UniTS supports only deterministic forecasting, when integrated with RL methods, we modify and retrain its output layer to produce predictive distributions; further details are in Appendix "Probabilistic Adaptation of UniTS". The compared post-training methods include full-parameter supervised fine-tuning (Full SFT), LoRA, MSFT (Qiao et al. 2025), TPO (Qi et al. 2025), TimeRFT (Li et al. 2026), and our proposed TS-GRPO. Implementation details of TS-GRPO are provided in Appendix "TS-GRPO". TPO is applicable only to TSFMs that adopt the one-shot forecasting paradigm, whereas TimeRFT is specifically designed for Moirai-MoE. Accordingly, we apply TPO only to Moirai and UniTS, and TimeRFT only to Moirai-MoE. In contrast, TS-GRPO can be applied to any TSFM with probabilistic outputs. Moreover, because TimeRFT involves a complex training pipeline and numerous hyperparameters, we compare against it only on the Loop Seattle and ENTSO-e Load datasets, for which its training scripts are publicly available. We do not compare with conventional TSF methods, such as iTransformer (Liu et al. 2023), PatchTST (Nie et al. 2023), DLinear (Zeng et al. 2023), TimeMixer (Wang et al. 2024), and SimpleTM (Chen et al. 2025), because pretrained TSFMs already achieve comparable performance, making comparisons with post-trained TSFMs unfair.

Implementation Details:

All experiments are implemented in PyTorch (Imambi, Prakash, and Kanagachidambaresan 2021) on an NVIDIA RTX 4090 GPU. We use the Adam optimizer (Kingma and Ba 2014). The hyperparameters λ_m and λ_u are both set to 1. The neighborhood radius r is set to 1 for Loop Seattle, 100 for ENTSO-e Load, and 0.1 for all other datasets. These values are selected based on the hyperparameter sensitivity results. Other settings, including the learning rate and batch size, follow the best configurations of the corresponding post-training methods. The maximum number of training epochs is set to 40. Early stopping is triggered when the validation loss does not decrease for three consecutive evaluation rounds.

Forecasting Performance

We evaluate GTN-R under both standard and zero-shot forecasting settings. In the standard setting, we compare our method with other methods, with results reported in Tables 1 and 2. In the zero-shot setting, the model is trained on one dataset and evaluated on another, with results shown in Table 3. All results are averaged over five runs with different random seeds. For each base method, the best post-training result is highlighted in bold, while the second-best is underlined. Overall, incorporating GTN-R can improve the performance of existing RL post-training methods.

Method Analysis

In this subsection, we further validate the effectiveness of the proposed method by visualizing how the model's output distribution evolves during training after incorporating GTN-R. The experimental setup and the selection of samples for visualization are consistent with those in Fig.2b. During training, we record the following three metrics: 1) the mean log probability assigned by the model's output distribution to the ground-truth neighborhoods of the selected training samples; 2) the sampling variance of the predictive trajectories for these samples; and 3) the forecasting performance of the trained model on the test set. We compare three settings: standard training, directly adding the ground-truth trajectory to the sampled group, and training with GTN-R. The results are presented in Figs.5. As shown in Fig.5a, GTN-R prevents the ground-truth-neighborhood probability mass from decreasing, thereby mitigating suboptimal collapse. Fig.5b further shows that its sampling variance declines much less than in the other two settings, indicating better preservation of the exploration ability. Consequently, GTN-R achieves superior test-set forecasting performance, as shown in Fig.5c.

Furthermore, we visualize the evolution of the predictive distributions for the training samples in Fig.1 after incorporating GTN-R. We use the same random seed in the experiments to ensure consistent training dynamics. The results are

Method	ETTm1		ETTm2		ETTh1		ETTh2		ECL		Weather	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Moirai-small	0.448	0.409	0.300	0.341	0.416	0.427	0.354	0.381	0.233	0.320	0.268	0.279
+ *Full finetuning*	0.367	0.382	0.273	0.316	0.415	0.428	0.352	0.378	0.193	0.279	0.228	0.254
+ *LoRA*	0.370	0.383	0.272	0.314	0.414	0.427	0.353	0.380	0.192	0.279	0.225	0.252
+ *MSFT (2025)*	0.353	0.377	0.250	0.301	0.412	0.426	0.349	0.375	0.187	0.275	0.215	0.248
+ *TPO (2025)*	0.361	0.376	0.268	0.310	0.410	0.424	0.348	0.374	0.191	0.276	0.225	0.251
+ *TPO (2025)* + ***GTN-R***	0.354	0.373	0.258	0.303	0.406	0.421	0.344	0.371	**0.179**	**0.267**	0.216	0.244
+ *TS-GRPO*	0.358	0.375	0.265	0.308	0.416	0.429	0.351	0.377	0.200	0.287	0.226	0.252
+ *TS-GRPO* + ***GTN-R***	**0.345**	**0.369**	**0.244**	**0.293**	**0.403**	**0.419**	**0.339**	**0.367**	**0.179**	**0.267**	**0.210**	**0.239**
Moirai-base	0.382	0.388	0.281	0.326	0.412	0.424	0.356	0.388	0.188	0.274	0.246	0.265
+ *Full finetuning*	0.368	0.371	0.258	0.307	0.408	0.424	0.357	0.385	0.173	0.264	0.232	0.258
+ *LoRA*	0.361	0.370	0.259	0.307	0.408	0.423	0.356	0.387	0.172	0.263	0.230	0.260
+ *MSFT (2025)*	**0.332**	0.369	0.247	0.305	0.407	0.422	0.352	0.383	0.169	0.260	**0.213**	0.244
+ *TPO (2025)*	0.365	0.368	0.256	0.304	0.402	0.418	0.352	0.380	0.171	0.261	0.229	0.255
+ *TPO (2025)* + ***GTN-R***	0.355	0.359	0.248	0.296	**0.390**	**0.403**	0.345	0.372	0.164	0.251	0.224	0.249
+ *TS-GRPO*	0.357	0.364	0.253	0.301	0.415	0.429	0.349	0.378	0.179	0.267	0.222	0.248
+ *TS-GRPO* + ***GTN-R***	0.343	**0.347**	**0.238**	**0.286**	**0.390**	**0.403**	**0.337**	**0.363**	**0.163**	**0.249**	0.217	**0.243**
Toto	0.396	0.378	0.267	0.303	0.435	0.413	0.340	0.363	0.161	0.243	0.224	0.245
+ *Full finetuning*	0.376	0.370	0.258	0.301	0.428	0.409	0.339	0.361	0.159	0.240	0.222	0.243
+ *LoRA*	0.378	0.374	0.260	0.301	0.426	0.407	0.338	0.360	0.158	0.239	0.221	0.242
+ *MSFT (2025)*	0.357	0.367	0.249	0.296	0.422	0.406	0.337	0.358	0.155	0.236	0.217	0.240
+ *TS-GRPO*	0.358	0.362	0.248	0.293	0.418	0.403	0.334	0.355	0.155	0.236	0.216	0.239
+ *TS-GRPO* + ***GTN-R***	**0.343**	**0.349**	**0.237**	**0.284**	**0.409**	**0.399**	**0.329**	**0.351**	**0.152**	**0.233**	**0.210**	**0.235**
UniTS	0.713	0.552	0.321	0.355	0.527	0.491	0.405	0.417	0.432	0.488	0.291	0.313
+ *Full finetuning*	0.395	0.405	0.296	0.338	0.442	0.435	0.386	0.408	0.214	0.283	0.257	0.282
+ *LoRA*	0.393	0.405	0.296	0.338	0.437	0.434	0.384	0.407	0.188	0.281	0.250	0.278
+ *MSFT (2025)*	0.390	0.403	0.286	0.333	0.434	0.430	0.380	0.405	0.184	0.279	0.241	0.272
+ *TPO (2025)*	0.386	0.399	0.283	0.329	0.429	0.426	0.376	0.401	0.182	0.276	0.239	0.270
+ *TPO (2025)* + ***GTN-R***	0.381	0.396	0.280	0.326	0.423	0.421	0.370	0.397	0.180	0.274	0.236	0.267
+ *TS-GRPO*	0.386	0.399	0.279	0.327	0.431	0.428	0.373	0.398	0.180	0.274	0.238	0.269
+ *TS-GRPO* + ***GTN-R***	**0.374**	**0.391**	**0.275**	**0.323**	**0.414**	**0.416**	**0.364**	**0.393**	**0.177**	**0.271**	**0.232**	**0.264**

Table 1: Forecasting results averaged from four prediction lengths $\in \{96, 192, 336, 720\}$. Most baseline results are from (Qiao et al. 2025); the rest are obtained by running the source code on the datasets.

Method	5%		20%		50%		100%	
	Loop Seattle							
	MSE ($\times 10^1$)	MAE ($\times 10^0$)	MSE ($\times 10^1$)	MAE ($\times 10^0$)	MSE ($\times 10^1$)	MAE ($\times 10^0$)	MSE ($\times 10^1$)	MAE ($\times 10^0$)
$MOIRAI - MoE_s$	4.186	3.971	4.186	3.971	4.186	3.971	4.186	3.971
+ TimeSFT	2.996	3.625	2.614	3.553	2.535	3.379	2.421	3.448
+ TimeLoRA	3.011	3.622	2.697	3.584	2.565	3.370	2.434	3.464
+ TS-GRPO	2.911	3.601	2.424	3.428	2.314	3.302	2.254	3.265
+ TS-GRPO + **GTN-R**	2.852	3.553	2.354	3.366	2.248	3.251	2.213	3.224
+ TimeRFT	2.757	3.400	2.193	3.091	2.082	3.019	2.032	3.021
+ TimeRFT + **GTN-R**	**2.702**	**3.357**	**2.125**	**3.012**	**2.013**	**2.956**	**1.994**	**2.978**
	ENTSO-e Load							
	MSE ($\times 10^5$)	MAE ($\times 10^2$)	MSE ($\times 10^5$)	MAE ($\times 10^2$)	MSE ($\times 10^5$)	MAE ($\times 10^2$)	MSE ($\times 10^5$)	MAE ($\times 10^2$)
$MOIRAI - MoE_s$	22.832	11.876	22.832	11.876	22.832	11.876	22.832	11.876
+ TimeSFT	16.066	8.860	6.465	5.561	6.682	5.800	5.379	5.232
+ TimeLoRA	15.801	8.836	6.291	5.498	6.770	5.869	5.348	5.209
+ TS-GRPO	16.146	8.941	6.520	5.621	4.312	4.619	4.095	4.532
+ TS-GRPO + **GTN-R**	16.013	8.852	6.452	5.560	4.237	4.571	3.999	4.412
+ TimeRFT	15.329	8.643	4.196	4.525	3.867	4.189	3.767	4.230
+ TimeRFT + **GTN-R**	**15.156**	**8.593**	**4.132**	**4.469**	**3.734**	**4.115**	**3.710**	**4.187**

Table 2: More Forecasting results. The baseline results derive from TimeRFT (Li et al. 2026). The percentages in the table indicate the proportion of the training data used. The experimental setup follows Table 2 in TimeRFT.

in Fig. 6. Comparing Figs.1 and 6, we observe that GTN-R mitigates suboptimal collapse as training progresses.

Conclusion

This paper identifies suboptimal collapse in RL post-training of TSFMs and analyzes its self-reinforcing mechanism. To mitigate its negative impact on training, we propose GTN-R, which increases the probability mass assigned to the ground-truth neighborhood while maintaining a uniform distribution within it. This alleviates suboptimal collapse while preserving the model's exploration capability. Experiments across multiple models, datasets, and RL methods demonstrate that GTN-R consistently improves forecasting performance.

Method	ETTm1→ETTm2		ETTm2→ETTm1		ETTm2→ETTh2		ETTh1→Weather	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Moirai-small	0.300	0.341	0.448	0.409	0.354	0.381	0.268	0.279
+ *Full finetuning*	0.293	0.339	0.453	0.412	0.358	0.384	0.263	0.276
+ *LoRA*	0.321	0.396	0.452	0.413	0.359	0.384	0.264	0.277
+ *MSFT (2025)*	0.288	0.336	0.469	0.419	0.355	0.381	0.259	0.272
+ *TPO (2025)*	0.290	0.335	0.443	0.407	0.354	0.380	0.261	0.274
+ *TPO (2025)* + ***GTN-R***	**0.278**	**0.332**	**0.437**	**0.403**	0.349	0.376	0.257	0.270
+ *TS-GRPO*	0.289	0.335	0.446	0.407	0.351	0.381	0.259	0.273
+ *TS-GRPO* + ***GTN-R***	0.280	0.334	0.438	0.404	**0.346**	**0.374**	**0.256**	**0.269**
Moirai-base	0.281	0.326	0.382	0.388	0.356	0.388	0.246	0.265
+ *Full finetuning*	0.274	0.324	0.387	0.391	0.360	0.391	0.242	0.262
+ *LoRA*	0.300	0.377	0.384	0.389	0.360	0.393	0.242	0.262
+ *MSFT (2025)*	0.269	0.320	0.400	0.397	0.358	0.390	0.238	0.258
+ *TPO (2025)*	0.271	0.324	0.378	0.385	0.356	0.386	0.238	0.258
+ *TPO (2025)* + ***GTN-R***	0.265	0.315	**0.371**	**0.380**	0.351	0.380	**0.234**	**0.254**
+ *TS-GRPO*	0.270	0.322	0.378	0.386	0.354	0.386	0.239	0.259
+ *TS-GRPO* + ***GTN-R***	**0.263**	**0.314**	0.373	0.382	**0.345**	**0.375**	0.236	0.256

Table 3: Zero-shot results averaged over prediction lengths $96, 192, 336, 720$.

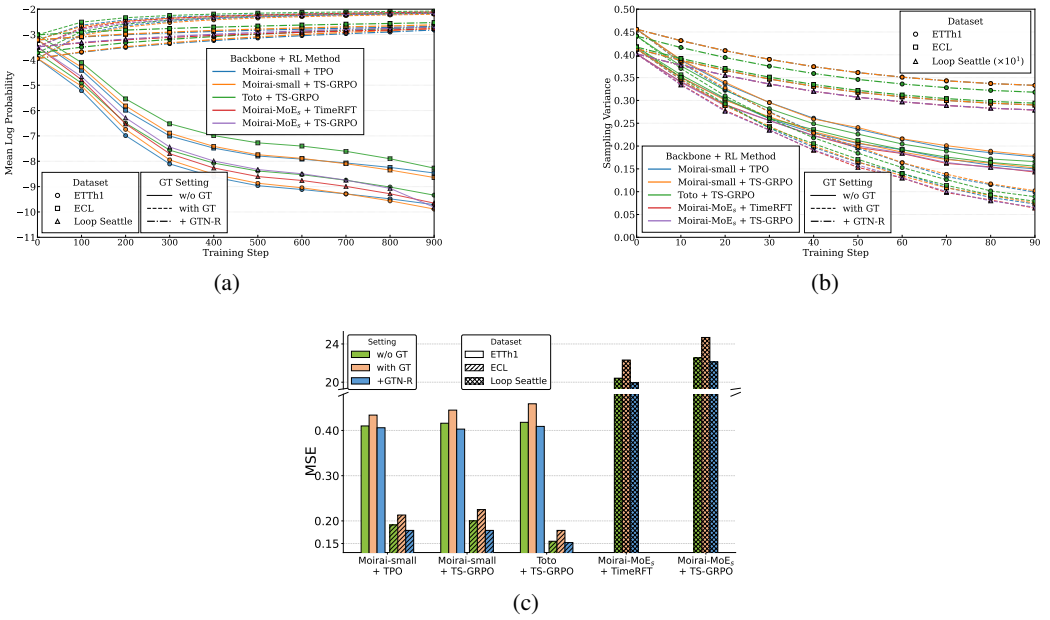

Figure 5: (a) Mean ground-truth-neighborhood log probability during training. (b) and sampling variance during training. (c) Test-set forecasting performance. In (b), $\times 10^1$ indicates that the actual variances of the dataset are 10 times the plotted values.

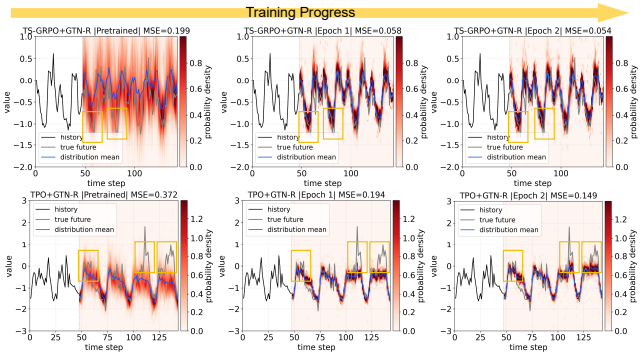

Figure 6: The output distributions for the training samples in Fig.1 after incorporating GTN-R. More visualization results are provided in Appendix "More Visualization".

References

Abhishek, K.; Singh, M.; Ghosh, S.; and Anand, A. 2012. Weather forecasting model using artificial neural network. *Procedia Technology*, 4: 311–318.

Boussif, O.; Boukachab, G.; Assouline, D.; Massaroli, S.; Yuan, T.; Benabbou, L.; and Bengio, Y. 2024. Improving* day-ahead* Solar Irradiance Time Series Forecasting by Leveraging Spatio-Temporal Context. *Advances in Neural Information Processing Systems*, 36.

Campbell, S. D.; and Diebold, F. X. 2005. Weather forecasting for weather derivatives. *Journal of the American Statistical Association*, 100(469): 6–16.

Chen, H.; Luong, V.; Mukherjee, L.; and Singh, V. 2025. SimpleTM: A simple baseline for multivariate time series forecasting. In *The Thirteenth International Conference on Learning Representations*.

Chu, T.; Zhai, Y.; Yang, J.; Tong, S.; Xie, S.; Schuurmans, D.; Le, Q. V.; Levine, S.; and Ma, Y. 2025. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*.

Clark, J. S.; and Bjørnstad, O. N. 2004. Population time series: process variability, observation errors, missing values, lags, and hidden states. *Ecology*, 85(11): 3140–3150.

Cohen, B.; Khwaja, E.; Wang, K.; Masson, C.; Ramé, E.; Doubli, Y.; and Abou-Amal, O. 2024. Toto: Time series optimized transformer for observability. *arXiv preprint arXiv:2407.07874*.

Das, A.; Kong, W.; Sen, R.; and Zhou, Y. 2024. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*.

Deb, C.; Zhang, F.; Yang, J.; Lee, S. E.; and Shah, K. W. 2017. A review on time series forecasting techniques for building energy consumption. *Renewable and Sustainable Energy Reviews*, 74: 902–924.

Fang, Y.; Qin, Y.; Luo, H.; Zhao, F.; and Zheng, K. 2023. STWave+: A Multi-Scale Efficient Spectral Graph Attention Network With Long-Term Trends for Disentangled Traffic Flow Forecasting. *IEEE Transactions on Knowledge and Data Engineering*.

Gao, S.; Koker, T.; Queen, O.; Hartvigsen, T.; Tsiligkaridis, T.; and Zitnik, M. 2024. Units: A unified multi-task time series model. *Advances in Neural Information Processing Systems*, 37: 140589–140631.

Goswami, M.; Szafer, K.; Choudhry, A.; Cai, Y.; Li, S.; and Dubrawski, A. 2024. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*.

Hamilton, L. C.; Brown, C.; and Keim, B. D. 2007. Ski areas, weather and climate: time series models for New England case studies. *International Journal of Climatology*.

Imambi, S.; Prakash, K. B.; and Kanagachidambaresan, G. 2021. PyTorch. In *Programming with TensorFlow: solution for edge computing applications*, 87–104. Springer.

Karevan, Z.; and Suykens, J. A. 2020. Transductive LSTM for time-series prediction: An application to weather forecasting. *Neural Networks*, 125: 1–9.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kottapalli, S. R. K.; Hubli, K.; Chandrashekhara, S.; Jain, G.; Hubli, S.; Botla, G.; and Doddaiah, R. 2025. Foundation models for time series: A survey. *arXiv preprint arXiv:2504.04011*.

Lara-Benítez, P.; Carranza-García, M.; Luna-Romera, J. M.; and Riquelme, J. C. 2020. Temporal convolutional networks applied to energy-related time series forecasting. *applied sciences*, 10(7): 2322.

Li, R.; Zhang, F.; Li, T.; Zhang, N.; and Zhang, T. 2022. DMGAN: Dynamic multi-hop graph attention network for traffic forecasting. *IEEE Transactions on Knowledge and Data Engineering*.

Li, S.; Chen, Y.; Zhu, Z.; Pan, Y.; Guo, Y.; Huang, M.; and Xiong, H. 2026. TimeRFT: Stimulating Generalizable Time Series Forecasting for TSFMs via Reinforcement Finetuning. *arXiv preprint arXiv:2605.00015*.

Liu, X.; Aksu, T.; Liu, J.; Wen, Q.; Liang, Y.; Xiong, C.; Savarese, S.; Sahoo, D.; Li, J.; and Liu, C. 2025. Empowering time series analysis with synthetic data: A survey and outlook in the era of foundation models. *arXiv preprint arXiv:2503.11411*.

Liu, X.; Liu, J.; Woo, G.; Aksu, T.; Liang, Y.; Zimmermann, R.; Liu, C.; Savarese, S.; Xiong, C.; and Sahoo, D. 2024a. Moirai-moe: Empowering time series foundation models with sparse mixture of experts. *arXiv preprint arXiv:2410.10469*.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2023. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*.

Liu, Y.; Zhang, H.; Li, C.; Huang, X.; Wang, J.; and Long, M. 2024b. Timer: Generative pre-trained transformers are large time series models. *arXiv preprint arXiv:2402.02368*.

Liu, Z.; Li, B.; Huang, H.; Sun, Y.; Wang, Y.; Wu, M.; and Ma, Q. 2026. From Pre-training to Post-training: A Survey on Time Series Foundation Models.

Miller, J. A.; Aldosari, M.; Saeed, F.; Barna, N. H.; Rana, S.; Arpinar, I. B.; and Liu, N. 2024. A survey of deep learning and foundation models for time series forecasting. *arXiv preprint arXiv:2401.13912*.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. *ICLR*.

Novo, R.; Marocco, P.; Giorgi, G.; Lanzini, A.; Santarelli, M.; and Mattiazzo, G. 2022. Planning the decarbonisation of energy systems: The importance of applying time series clustering to long-term models. *Energy Conversion and Management: X*, 15: 100274.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Qi, Y.; Hu, H.; Lei, D.; Zhang, J.; Shi, Z.; Huang, Y.; Chen, Z.; Lin, X.; and Shen, Z.-J. M. 2025. Timehf: Billion-scale time series models guided by human feedback. *arXiv preprint arXiv:2501.15942*.

Qiao, Z.; Liu, C.; Zhang, Y.; Jin, M.; Pham, Q.; Wen, Q.; Suganthan, P.; Jiang, X.; and Ramasamy, S. 2025. Multi-Scale Finetuning for Encoder-based Time Series Foundation Models. *arXiv preprint arXiv:2506.14087*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shekhar, S.; and Williams, B. M. 2007. Adaptive seasonal time series models for forecasting short-term traffic flow. *Transportation Research Record*, 2024(1): 116–125.

Swarztrauber, P. N. 2003. On computing the points and weights for Gauss–Legendre quadrature. *SIAM Journal on Scientific Computing*, 24(3): 945–954.

Wang, H.; Chen, J.; Fan, Z.; Zhang, Z.; Cai, Z.; and Song, X. 2022. St-expertnet: A deep expert framework for traffic prediction. *IEEE Transactions on Knowledge and Data Engineering*.

Wang, S.; Shen, Y.; Sun, H.; Feng, S.; Teng, S.-H.; Dong, L.; Hao, Y.; and Chen, W. 2025. Benefits and pitfalls of reinforcement learning for language model planning: a theoretical perspective. *arXiv preprint arXiv:2509.22613*.

Wang, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J. Y.; and Zhou, J. 2024. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*.

Weltevrede, M.; Kaubek, F.; Spaan, M. T.; and Böhmer, W. 2024. Explore-Go: Leveraging Exploration for Generalisation in Deep Reinforcement Learning. *arXiv preprint arXiv:2406.08069*.

Woo, G.; Liu, C.; Kumar, A.; Xiong, C.; Savarese, S.; and Sahoo, D. 2024. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*.

Yan, Y.; Tang, K.; Chen, S.; Xu, K.; Hu, D.; Yu, Q.; and Hu, P. 2026. S-GRPO: Unified Post-Training for Large Vision-Language Models. *arXiv preprint arXiv:2604.16557*.

Ye, J.; Yu, Y.; Zhang, W.; Wang, L.; Li, J.; and Tsung, F. 2026. Empowering time series analysis with foundation models: A comprehensive survey. *Information Fusion*, 104601.

Yoon, T.; Park, Y.; Ryu, E. K.; and Wang, Y. 2022. Robust probabilistic time series forecasting. In *International Conference on Artificial Intelligence and Statistics*, 1336–1358. PMLR.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are Transformers Effective for Time Series Forecasting? *AAAI*.