# OATS: Online Data Augmentation for Time Series Foundation Models

**Junwei Deng**[*]
University of Illinois Urbana-Champaign

**Chang Xu**[†]
Microsoft Research

**Jiaqi Ma**
University of Illinois Urbana-Champaign

**Jiang Bian**
Microsoft Research

## Abstract

Data augmentation is a key approach for enhancing the training of Time Series Foundation Models (TSFM). However, existing augmentation methods typically rely on heuristic synthetic data generation and follow a static paradigm, where synthetic data is generated once prior to training. To overcome these limitations, we propose Online Data Augmentation for Time Series Foundation Models (OATS), an algorithm to generate high-quality synthetic time series data through principled and dynamically adaptive augmentation strategies. OATS introduces methods to identify high-quality samples, guide the generation process with these samples, and scale augmentation efficiently. Experiments on TSFM demonstrate that OATS significantly improves the generalization performance of TSFM pretraining.

## 1 Introduction

Time series modeling plays a critical role across a wide range of domains, including finance [Kim et al., 2019, Li et al., 2024], healthcare [Guo et al., 2023], climate science [Liang et al., 2023], and industrial monitoring [Zamanzadeh Darban et al., 2024]. Traditionally, time series models have been trained on relatively small and single-domain datasets which are often carefully curated [Wen et al., 2022]. However, recent developments such as time series foundation models (TSFM) [Yao et al., 2024, Ansari et al., 2024] lead to a transition toward large-scale datasets across different domains. These training datasets are often curated through multiple third-party time series data sources [Yao et al., 2024]. Typical issues in time series datasets include missing values [Junninen et al., 2004], heterogeneous sampling rates [Woo et al., 2024], imbalanced domain distributions [Yao et al., 2024], data duplication [Lin et al., 2023], and so on. Nevertheless, **the success of time series foundation models relies on the availability of high-quality and diverse data**. As the time series community shifts toward large-scale and foundation models, optimizing training data has become one of the most pressing challenges in time series modeling.

***Data augmentation* with synthetic data has emerged as a key alternative for optimizing time series training data in current TSFM studies.** Many TSFMs [Woo et al., 2024, Das et al., 2024, Liu et al., 2024] construct large-scale and diverse pretraining datasets by manually collecting data from multiple sources across different domains. Beyond collecting real-world data, data augmentation with synthetic data is also widely applied to further improve the training data [Das et al., 2024, Ansari et al., 2024]. However, the role and usefulness of data augmentation remain debatable, with conflicting findings reported across different studies [Fu et al., 2024, Kuvshinova et al., 2024, Ansari et al., 2024, Deng et al., 2025]. **We identify two fundamental limitations in current data augmentation**

[*]Work done during research internship at Microsoft Research.

[†]Corresponding Author: chanx@microsoft.com.

NeurIPS 2025 Workshop on Recent Advances in Time Series Foundation Models (BERT$^2$S).

**techniques for TSFMs that may undermine their effectiveness. First**, most existing methods rely on heuristic strategies for generating synthetic data. These approaches are typically rule-based or guided by human intuition about quality constraints such as smoothness, periodicity, or trend structure that synthetic time series are expected to resemble plausible real-world data. Despite some efforts [Shi et al., 2024, Ansari et al., 2024, Das et al., 2024] in proposing different synthetic data generation methods, there remains a pressing need for more principled criteria to assess the quality of synthetic time series. **Second**, the generation mostly follows a static paradigm, which means the quality constraints are set and fixed for the whole training process. This fails to account for the evolving learning dynamics of the model and may miss opportunities to adaptively refine the augmented dataset based on the model's training progression. Recent studies suggest that data quality is closely tied to training dynamics, highlighting the potential benefits of more adaptive, model-aware augmentation strategies [Wang et al., 2024b,a].

To address these two limitations, we propose OATS (Online Data Augmentation for Time Series Foundation Models), an algorithm designed to enable dynamic data augmentation for TSFMs. OATS consists of three core components: **Time-series influence scores (TSIS)** integrate training data attribution with time series–specific knowledge to dynamically assess the quality of each data sample in a principled manner. **High-quality guided data augmentation** leverages top-ranked samples in TSIS as prompts to condition a diffusion model for synthetic data generation. To reduce computational overhead and effectively balance between leveraging historical knowledge and exploring new samples, OATS adopts an **explore–exploit paradigm**. Specifically, the influence scores are periodically re-evaluated to incorporate model training dynamics while preserving previously identified high-quality data. Finally, we conduct experiments of the dataset augmentation carried out by OATS, demonstrating the effectiveness and practical benefits of our approach.

## 2 Method

In this section, we introduce the modules of Online Data Augmentation for Time Series Foundation Models (OATS). The proposed method acts as a systematical approach to incorporating synthetic data in the training process. OATS involves three modules, i.e., *Time-series Influence Scores (TSIS)* to quantitatively estimate the quality of time series samples, *High-quality guided Data Augmentation* to generate synthetic samples with high-quality samples as prompts, and the *explore-exploit paradigm* to balance the efficiency and effectiveness. We will introduce the modules in separate paragraphs. A diagram of the whole algorithm is presented in Figure 1 and the algorithm block is shown in Appendix A.

**Notations.** Suppose we have a training dataset $\mathcal{D}_{tr}$ of size $N$ which is partitioned into $L$ disjoint subsets: $\mathcal{D}_{tr} = \bigcup_{l=1}^{L} \mathcal{D}_l = \{z_{l,k} | l = [L]; k = [N_l]\}$, where $N_l = |D_l|$ is the size of subset $\mathcal{D}_l$ and $z_{l,k} \in \mathcal{Z}$, the data space of time series samples. A TSFM parameterized by $w \in \mathcal{W}$ is being trained on $\mathcal{D}_{tr}$ to minimize loss function $\ell$ via an iterative optimization algorithm, e.g., stochastic gradient descent [Ruder, 2016] for $T$ steps. The intermediate checkpoints of each step are represented as $\{w_t | t = [T]\}$. We also have a validation dataset $\mathcal{D}_{val} = \{z_v | v = [N_v]\}$, $z_v \in \mathcal{Z}$ and $\mathcal{D}_{val}$ is not overlapped with $\mathcal{D}_{tr}$.

**Time-series Influence Scores (TSIS).** In OATS, we employ *data attribution* [Koh and Liang, 2017, Pruthi et al., 2020] as a principled method to estimate the influence of a data point on the model output, which has shown significant usefulness in dataset optimization in different areas [Wang et al., 2024b, Xia et al., 2024]. We design the TSIS function with respect to a validation dataset $\mathcal{D}_{val}$ as $\mathcal{F}_{\mathcal{D}_{val}}$ for each sample to be $\mathcal{F}_{\mathcal{D}_{val}} : \mathcal{D}_{tr} \times \mathcal{W} \to \mathbb{R}$, which estimates the influence score of a data sample to be trained on $w_t$ with respect to the performance on $\mathcal{D}_{val}$. A larger score indicates that training on $z$ for the next training step $(t+1)$ leads to better performance on $\mathcal{D}_{val}$. We additionally consider TS-specific quality indicators, which we use signal-to-noise ratio (SNR). The TSIS function can be defined as:

$$\mathcal{F}_{\mathcal{D}_{val}}(z_i, w_t) = \underbrace{g_{(\mathrm{val});\mathrm{t}} \cdot g_{z_i;t}}_{\text{Influence Score}} + \underbrace{\lambda \cdot \mathrm{SNR}(z_i)}_{\text{TS-specific quality}}, \tag{1}$$

where $\lambda$ is the hyperparameter controls the significance of the time series specific quality term, $g_{(\mathrm{val});\mathrm{t}} = \nabla_w \sum_{v_i \in \mathcal{D}_{val}} \ell(v_i, w_t)/|\mathcal{D}_{val}|$ and $g_{z_i;t} = \nabla_w \ell(z_i, w_t)$.

**High-quality Guided Data Augmentation.** We design a high-quality data guidance generation model $\mathcal{G}$ for online synthetic data generation. The quality of data samples is indicated by TSIS we

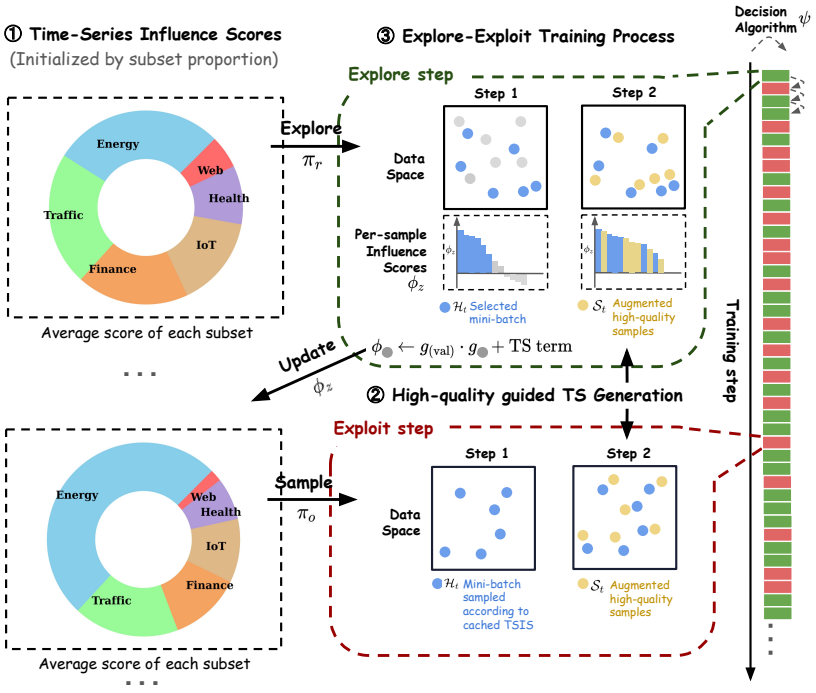



Figure 1: Architecture of OATS. Modules include ① Time-Series Influence Scores, ② High-Quality Guided Time Series Augmentation and ③ Explore–Exploit Training Process.

introduced. We aim to model the conditional distribution $p(\hat{z}|\mathcal{D}_{\text{high}})$, which defines the probability of generating a sample $\hat{z}$ given the high-quality dataset $\mathcal{D}_{\text{high}} \subseteq \mathcal{D}_{tr}$. The conditional generation model $\mathcal{G}$ serves as a parameterized approximation of this distribution, enabling practical sampling via $\hat{z} \sim \mathcal{G}(\mathcal{D}_{\text{high}})$.

We design the architecture of $\mathcal{G}$ to be a diffusion model utilizing a time series semantic prototype module and take $\mathcal{D}_{\text{high}}$ as a generation prompt, taking Huang et al. [2025] as our backbone model. The model extracts prototype weights $\mathcal{M} = \{m_i|z_i \in \mathcal{D}_{\text{high}}\}$ from prompt samples. We leverage the same design of weight extractor and conditional noise predictor as in Huang et al. [2025]. We additionally include a class conditional guidance based on the subset information, denoted as $\mathcal{C}_{\text{high}} = \{c_i|z_i \in \mathcal{D}_{\text{high}}\}$, where each $c_i \in [L]$ indicates which disjoint subset $\mathcal{D}_l$ the sample $z_i$ belongs to. The generation (reverse diffusion) for each prompt can be expressed as

$$p(z^{(t-1)}|z^{(t)}, \mathcal{D}_{\text{high}}) = \mathcal{N}(z^{(t-1)}|\mu(z^{(t)}, t, \underbrace{m_i, c_i}_{\text{High-quality data guidance}}, \sigma_t), \tag{2}$$

where $z^{(t)}$ is the intermediate sample at diffusion timestep $t$, $\mathcal{N}$ is Gaussian distribution, $\sigma_t$ is the noise covariance at diffusion step $t$. The architecture of the parameterized conditional noise predictor handles the prototype weights $m_i$ and class condition $c_i$.

**Explore & Exploit.** The TSIS we proposed is updated online through the training process and depends on every training step $w_t$. We are inspired by the multi-armed bandit problem and design an explore-exploit mechanism that could reuse the TSIS calculated for samples of previous training steps to reduce overhead.

The mechanism divided training steps into *explore* steps and *exploit* steps. In *explore* steps, we calculate TSIS for a batch of data samples $\mathcal{B}_t$ sampled by strategy $\pi_r$. We assume the locality of TSIS among $z$ in the same subset. Based on this assumption, we maintain a dynamic cache $\Phi_{\mathcal{D}_l} \in \mathbb{R}, l \in [L]$ for each subset and update it through exponentially moving average. The partition of $\mathcal{D}_{tr} = \bigcup_{l=1}^{L} \mathcal{D}_l$ can naturally be the sub-datasets of a large collection or some clustering results.

The update of $\Phi_{\mathcal{D}_l}$ on training step $t$ can be represented as

$$\Phi_{\mathcal{D}_l} = (1-\beta)\Phi_{\mathcal{D}_l} + \beta \sum_{z_i \in \mathcal{D}_l \cap \mathcal{B}_t} \mathcal{F}_{\mathcal{D}_{val}}(z_i, w_t)/|\mathcal{D}_l \cap \mathcal{B}_t| \tag{3}$$

where $\mathcal{B}_t \sim \pi_r^{|\mathcal{B}_t|}$, $\pi_r = \mathcal{U}(\mathcal{D}_{tr})$, $\beta \in [0,1]$ is the hyperparameter controls the decay factor of the exponentially moving average. For fast evolution of TSIS along training process, $\beta$ should be set larger and vice versa. The design of $\pi_r$ for explore steps reflects the spirit to visit new data points and update the TS data. It will randomly sample from the full training dataset $\mathcal{D}_{tr}$

For *exploit* step, we design a sample algorithm $\pi_o$ utilizing the $\Phi_{\mathcal{D}_l}$ cached in *explore* step. $\pi_o$ follows a two stage sampling strategy: 1) sample a subset $\mathcal{D}_l$ with probability $\frac{|\mathcal{D}_l|\cdot\max(0,\Phi_{\mathcal{D}_l})}{\sum_k |\mathcal{D}_k|\cdot\max(0,\Phi_{\mathcal{D}_k})}$ 2) uniformly sample a data point from the subset $\mathcal{D}_l$.

The choice of *explore* or *exploit* step is handled by $\epsilon$-greedy, a popular strategy for explore and exploit. We design a strategy $\psi$ is defined to be "explore" with probability $\epsilon$ and "exploit" with probability $1-\epsilon$, where $\epsilon$ is a hyperparameter controlling the balance between explore and exploit.

## 3 Experiments

We conduct experiments on Encoder-only TSFM, which is a modified version of Moirai [Woo et al., 2024], introduced by Yao et al. [2024]. We follow the same data processing and training settings as in Yao et al. [2024]. Detailed experiment settings can be found in Appendix B.

We evaluate 6 out-of-distribution datasets in LSF [Wu et al., 2023], and take a very small number of samples (32 in all our experiments) from these evaluation datasets as validation samples used by TSIS. We compare the performance of OATS and regular pretraining. "OATS (Sel+Aug)" is the full algorithm proposed in Section 2 and Algorithm 1, and "Regular" is the regular pretraining process used by Yao et al. [2024]. We also include a variant of OATS, i.e., "OATS (Sel only)", which only selects high-quality data points through TSIS while not carrying out data augmentation for the ablation study. Essentially, "OATS (Sel only)" is Algorithm 1 without "Step 2" in both explore and exploit steps.

In Table 1, we present the test loss (negative log likelihood, NLL) and MAPE on the evaluation datasets. It shows that "OATS (Sel+Aug)" outperforms other algorithms on most of the datasets (Weather, ETTh1, ETTm2, Electricity, ETTh1), and "OATS (Sel only)" shows superiority occasionally on some datasets and prediction lengths (ETTm1). The experiment shows that OATS effectively helps identify high-quality data samples from the training set and incorporate synthetic data samples guided by high-quality data that improve the TSFM pretraining performance.

Table 1: Performance of OATS ($\epsilon = 1$) and regular training of TSFM [Yao et al., 2024] pretraining.

| Dataset | Pred. length | OATS (Sel+Aug) | | OATS (Sel only) | | Regular | | Dataset | Pred. length | OATS (Sel+Aug) | | OATS (Sel only) | | Regular | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | NLL | MAPE | NLL | MAPE | NLL | MAPE | | | NLL | MAPE | NLL | MAPE | NLL | MAPE |
| ETTh1 | 96 | **1.754** | **0.684** | 1.774 | 0.878 | 1.866 | 0.823 | ETTh2 | 96 | **1.869** | **0.235** | 1.929 | 0.243 | 2.072 | 0.256 |
| | 192 | **1.756** | **0.705** | 1.758 | 0.796 | 1.853 | 0.751 | | 192 | **1.898** | **0.27** | 1.951 | 0.278 | 2.112 | 0.288 |
| | 336 | 1.829 | **0.82** | **1.82** | 0.861 | 1.912 | 0.835 | | 336 | **1.937** | **0.302** | 1.993 | 0.312 | 2.154 | 0.326 |
| | 720 | 1.851 | 0.996 | **1.826** | 0.969 | 1.916 | 0.964 | | 720 | **2.029** | **0.299** | 2.071 | 0.307 | 2.238 | 0.319 |
| ETTm1 | 96 | **1.569** | 0.54 | 1.579 | **0.518** | 1.816 | 0.698 | ETTm2 | 96 | **1.892** | **0.218** | 2.164 | 0.306 | 2.185 | 0.314 |
| | 192 | 1.647 | 0.669 | **1.645** | **0.583** | 1.877 | 0.861 | | 192 | **1.879** | **0.209** | 2.119 | 0.277 | 2.136 | 0.279 |
| | 336 | **1.669** | 0.643 | 1.68 | **0.607** | 1.864 | 0.798 | | 336 | **1.851** | **0.247** | 2.06 | 0.301 | 2.102 | 0.304 |
| | 720 | 1.704 | 0.651 | **1.699** | **0.62** | 1.866 | 0.781 | | 720 | **1.959** | **0.295** | 2.13 | 0.332 | 2.21 | 0.334 |
| Weather | 96 | **3.201** | **1.524** | 3.494 | 2.757 | 3.528 | 2.418 | Electricity | 96 | **5.887** | **0.403** | 6.134 | 0.676 | 6.18 | 0.742 |
| | 192 | **3.217** | **2.105** | 3.484 | 3.239 | 3.493 | 2.414 | | 192 | **5.965** | **0.473** | 6.183 | 0.659 | 6.233 | 0.751 |
| | 336 | **3.318** | **1.987** | 3.587 | 3.157 | 3.600 | 2.289 | | 336 | **5.984** | **0.536** | 6.263 | 0.71 | 6.276 | 0.817 |
| | 720 | **3.723** | 2.190 | 3.968 | 3.049 | 4.029 | **2.171** | | 720 | **6.078** | **0.544** | 6.377 | 0.721 | 6.398 | 0.829 |

Additional experiment results are included in Appendix C, where we show the test loss curve along the training process and the performance of OATS with $\epsilon \neq 1$.

## 4 Conclusion

In this paper, we introduce OATS, an algorithm to generate high-quality synthetic time series data through principled and dynamically adaptive augmentation strategies for TSFM. The key design

choice behind OATS is to incorporate high-quality synthetic data into the training process. We leverage data attribution methods to calculate the influence score as the primary indicator of data quality and data-driven diffusion models to generate new synthetic data samples conditional on the high-quality data samples. Empirical evaluations show that OATS could improve TSFM performance. By showing the advantage of dynamic and adaptive augmentation, OATS opens new avenues for data augmentation of TSFM.

## References


A. F. Ansari, L. Stella, C. Turkmen, X. Zhang, P. Mercado, H. Shen, O. Shchur, S. S. Rangapuram, S. Pineda Arango, S. Kapoor, J. Zschiegner, D. C. Maddix, M. W. Mahoney, K. Torkkola, A. Gordon Wilson, M. Bohlke-Schneider, and Y. Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=gerNCVqqtR.

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

B. Deng, C. Xu, H. Li, Y. Huang, M. Hou, and J. Bian. Tardiff: Target-oriented diffusion guidance for synthetic electronic health record time series generation. *arXiv preprint arXiv:2504.17613*, 2025.

F. Fu, J. Chen, J. Zhang, C. Yang, L. Ma, and Y. Yang. Are synthetic time-series data really not as good as real data? *arXiv preprint arXiv:2402.00607*, 2024.

L. L. Guo, E. Steinberg, S. L. Fleming, J. Posada, J. Lemmon, S. R. Pfohl, N. Shah, J. Fries, and L. Sung. Ehr foundation models improve robustness in the presence of temporal distribution shift. *Scientific Reports*, 13(1):3767, 2023.

Y.-H. Huang, C. Xu, Y. Wu, W.-J. Li, and J. Bian. Timedp: Learning to generate multi-domain time series with domain prompts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 17520–17527, 2025.

H. Junninen, H. Niska, K. Tuppurainen, J. Ruuskanen, and M. Kolehmainen. Methods for imputation of missing values in air quality data sets. *Atmospheric environment*, 38(18):2895–2907, 2004.

R. Kim, C. H. So, M. Jeong, S. Lee, J. Kim, and J. Kang. Hats: A hierarchical graph attention network for stock movement prediction. *arXiv preprint arXiv:1908.07999*, 2019.

P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

K. Kuvshinova, O. Tsymboi, A. Kostromina, D. Simakov, and E. Kovtun. Towards foundation time series model: To synthesize or not to synthesize? *arXiv preprint arXiv:2403.02534*, 2024.

J. Li, Y. Liu, W. Liu, S. Fang, L. Wang, C. Xu, and J. Bian. Mars: a financial market simulation engine powered by generative foundation model. *arXiv preprint arXiv:2409.07486*, 2024.

Y. Liang, Y. Xia, S. Ke, Y. Wang, Q. Wen, J. Zhang, Y. Zheng, and R. Zimmermann. Airformer: Predicting nationwide air quality in china with transformers. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 14329–14337, 2023.

S. Lin, B. Liu, J. Li, and X. Yang. Common diffusion noise schedules and sample steps are flawed. 2024 ieee. In *CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 5392–5399, 2023.

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

G. Pruthi, F. Liu, S. Kale, and M. Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930, 2020.

S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

X. Shi, S. Wang, Y. Nie, D. Li, Z. Ye, Q. Wen, and M. Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. *arXiv preprint arXiv:2409.16040*, 2024.

J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *CoRR*, abs/2010.02502, 2020. URL `https://arxiv.org/abs/2010.02502`.

J. T. Wang, P. Mittal, D. Song, and R. Jia. Data shapley in one training run. *arXiv preprint arXiv:2406.11011*, 2024a.

J. T. Wang, T. Wu, D. Song, P. Mittal, and R. Jia. Greats: Online selection of high-quality data for llm training in every iteration. *Advances in Neural Information Processing Systems*, 37: 131197–131223, 2024b.

Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

G. Woo, C. Liu, A. Kumar, C. Xiong, S. Savarese, and D. Sahoo. Unified training of universal time series forecasting transformers. 2024.

H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=ju_Uqw384Oq`.

M. Xia, S. Malladi, S. Gururangan, S. Arora, and D. Chen. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*, 2024.

Q. Yao, C.-H. H. Yang, R. Jiang, Y. Liang, M. Jin, and S. Pan. Towards neural scaling laws for time series foundation models. *arXiv preprint arXiv:2410.12360*, 2024.

Z. Zamanzadeh Darban, G. I. Webb, S. Pan, C. Aggarwal, and M. Salehi. Deep learning for time series anomaly detection: A survey. *ACM Computing Surveys*, 57(1):1–42, 2024.

# A Algorithm and Complexity Analysis

**Algorithm.** We propose Online Data Augmentation for Time Series Foundation Models (OATS) and present in Algorithm 1.

**Algorithm 1** Online Data Augmentation for Time Series Foundation Models (OATS)

**Require:** Training dataset and its $L$ disjoint subsets $\mathcal{D}_{tr} = \bigcup_{l=1}^{L} \mathcal{D}_l$, validation dataset $\mathcal{D}_{val}$, training batch size $b$, conditional augmentation algorithm $\mathcal{G}$, training loss $\ell$, explore-exploit strategy $\psi$, explore sampling algorithm $\pi_r$, exploit sampling algorithm $\pi_o$, TSIS function $\mathcal{F}_{\mathcal{D}_{val}}$.
Initialize model $w_0$
Initialize TSIS per subset $\Phi_{\mathcal{D}_l}, l \in [L]$ to be subset proportion.
**for** $t = 1$ to $T$ **do**
  **if** $\psi(t)$ == "explore" **then**
    Sample $\mathcal{B}_t \sim \pi_r^b$ ▷ Step 1
    Calculate $\mathcal{F}_{\mathcal{D}_{val}}(z_i, w_t)$ for $z_i \in \mathcal{B}_t$
    Update $\Phi_{\mathcal{D}_l}$ according to Equation 3.
    Select a subset $\mathcal{H}_t \subseteq \mathcal{B}_t$ of samples with top-$b/2$ of value $\mathcal{F}_{\mathcal{D}_{val}}(z_i, w_t)$.
    Generate $b/2$ samples as $\mathcal{S}_t$ using $\mathcal{G}$ guided by $\mathcal{H}_t$. ▷ Step 2
  **end if**
  **if** $\psi(t)$ == "exploit" **then**
    Sample $\mathcal{H}_t \sim \pi_o^{b/2}$ ▷ Step 1
    Generate $b/2$ samples as $\mathcal{S}_t$ using $\mathcal{G}$ guided by $\mathcal{H}_t$. ▷ Step 2
  **end if**
  Update $w_t$ on mini-batch data $\mathcal{H}_t \cup \mathcal{S}_t$ and get $w_{t+1}$, continue to next step $t+1$.
**end for**

**Complexity Analysis.** OATS involves computational overhead to the training process, which can be split into two sources: 1) calculation of TSIS and 2) synthetic data generation.

For TSIS calculation, the overhead of Equation 1 mainly comes from the gradient calculation of validation data samples in "explore" steps. In our experiment, we use a small number of validation data samples (32) that matches the training batch size. This helps maintain the same computation complexity between TSIS and the regular gradient descent training step as $\mathcal{O}(b)$, where $b$ is the training batch size. Furthermore, the overhead of "exploit" steps is small enough to be ignored since TSIS in Equation 1 is not used in those steps.

For synthetic data generation through diffusion model sampling, it is hard to compare the complexity with the regular gradient descent training step. Empirically, we find that some accelerated sampling strategies like DDIM [Song et al., 2020] could perform well enough with very small sample steps.

# B Experiment Settings

**Models.** We conduct the experiment on the TSFM models with the same configuration as the encoder-only model with a 10M parameter size in Yao et al. [2024]. The model is a modified version of Moirai [Woo et al., 2024], introduced by Yao et al. [2024], which incorporates patch embedding, rotary positional embedding, and a mixture of distributions to better adapt to time series forecasting while preserving extensibility.

**Training Process.** We adopt a similar training setting as in Yao et al. [2024] for the experiment. We utilize the AdamW optimizer with a batch size of 32 and a maximum learning rate of $10^{-3}$ with a linear warm-up of $10^4$ training steps, followed by cosine decay for the remaining $2 \times 10^4$ steps.

**Datasets for TSFM.** We pretrain the TSFM on the modified (balanced domain sample, quality filtering) LOTSA-100M dataset in the pretraining stage provided by Yao et al. [2024]. We take the native sub-dataset division as subsets in our experiment. These models are then evaluated on the out-of-distribution LSF dataset [Wu et al., 2023], using various prediction lengths (96, 192, 336, 720) and the same preprocessing pipeline as in Yao et al. [2024]. The detailed information of evaluation datasets is stated in Table 2.

Table 2: Evaluation datasets and properties.

| Dataset | Domain | Frequency | # Prediction Length |
|---|---|---|---|
| ETTh1 | Energy | H | 96/192/336/720 |
| ETTh2 | Energy | H | 96/192/336/720 |
| ETTm1 | Energy | 15min | 96/192/336/720 |
| ETTm2 | Energy | 15min | 96/192/336/720 |
| Electricity | Energy | H | 96/192/336/720 |
| Weather | Climate | H | 96/192/336/720 |

**Datasets for Generation Model.** We train the generation model for high-quality guided data augmentation described in Section 2 by a sampled dataset from the training dataset of TSFM. We sample 5% of the dataset in 20 selected subsets (Table 3) in LOTSA-100M as the training set of the diffusion model. The generation length of the diffusion model is set to 320, and the diffusion step is set to 200. We use DDIM for the sampling process with 40 steps.

Table 3: Training datasets of generation model.

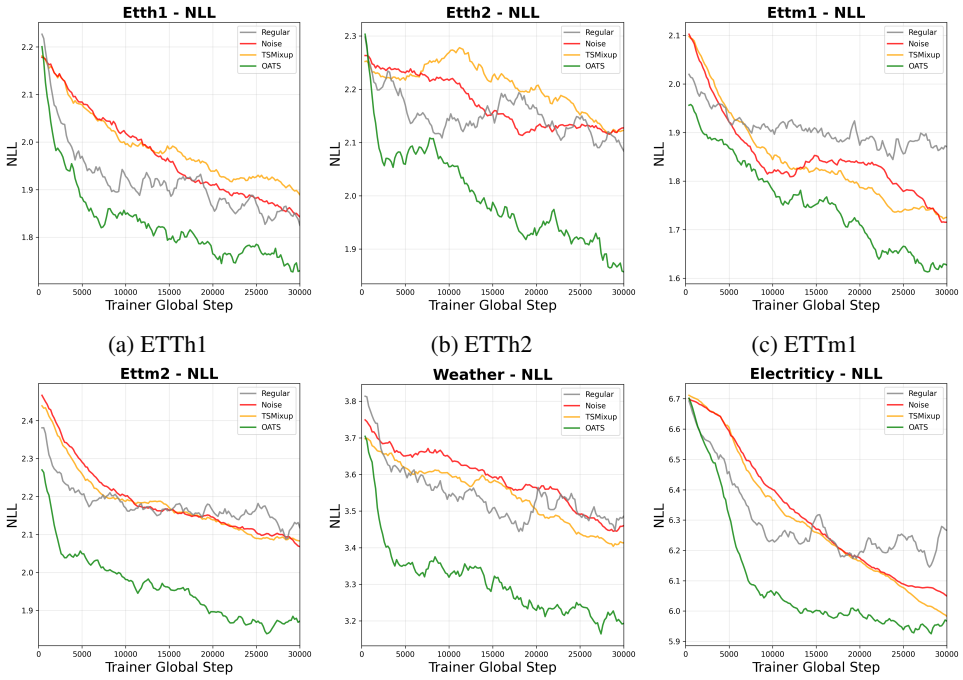
| Dataset | Domain | Frequency |
|---|---|---|
| CMIP6 | Climate | 6H |
| ERA5 | Climate | H |
| CloudOpsTSF | CloudOTS | 5T |
| Azure VM Traces 2017 | CloudOTS | 5T |
| Loop Seattle | Transport | 5T |
| PEMS07 | Transport | 5T |
| PEMS Bay | Transport | 5T |
| Q-Traffic | Transport | 15T |
| Largest 2017 | Transport | 5T |
| Largest 2018 | Transport | 5T |
| Largest 2019 | Transport | 5T |
| Largest 2020 | Transport | 5T |
| Largest 2021 | Transport | 5T |
| Australian Electricity | Energy | 30T |
| Buildings900K | Energy | H |
| Solar Power | Energy | 4S |
| Favorita Sales | Sales | D |
| Wiki-Rolling | Web | D |
| LibCity | Transport | 5T |
| OthersLOTSA | Energy | H |

**Metrics.** To be consistent with Yao et al. [2024], we primarily report the normalized mean absolute percentage error (MAPE) and negative log-likelihood (NLL), as these metrics avoid distortions caused by high-amplitude samples.

## C Additional Experiment Results

In Figure 2, the test loss along the training process is shown for the evaluation datasets. The curve is smoothed using time-weighted EMA. As shown in the figure, both `OATS` (Sel+Aug) (in green) and `OATS` (Sel only) (in blue) with only data selection significantly outperform the regular pretraining (in grey).

In Table 4, we present results of `OATS` with various $\epsilon$ on the ETTh1 dataset. The result shows that even if we set $\epsilon = 0.5$, most of the metrics still outperform regular training, which shows a great potential to reduce the `OATS` overhead without hurting the performance too much.




(a) ETTh1 (b) ETTh2 (c) ETTm1

(d) ETTm2 (e) Weather (f) Electricity

Figure 2: Comparison of different algorithms on test loss (NLL) during the training.

Table 4: Performance of various $\epsilon$ on ETTh1 dataset. **bold** represent the best result and underline represent OATS ($\epsilon$) outperforms regular training.

| Pred. Length | $\epsilon$=1 | | $\epsilon$=0.7 | | $\epsilon$=0.5 | | Regular | |
|---|---|---|---|---|---|---|---|---|
| | NLL | MAPE | NLL | MAPE | NLL | MAPE | NLL | MAPE |
| 96 | **1.754** | **0.684** | 1.774 | 0.713 | 1.818 | 0.785 | 1.866 | 0.823 |
| 192 | **1.756** | 0.705 | 1.771 | **0.693** | 1.812 | 0.739 | 1.853 | 0.751 |
| 336 | **1.829** | 0.82 | 1.833 | **0.759** | 1.874 | 0.855 | 1.912 | 0.835 |
| 729 | **1.851** | 0.996 | 1.861 | 0.946 | 1.890 | 0.999 | 1.916 | **0.964** |