Timer: Generative Pre-trained Transformers Are Large Time Series Models

Yong Liu [*1] Haoran Zhang [*1] Chenyu Li [*1] Xiangdong Huang [1] Jianmin Wang [1] Mingsheng Long [1]

Abstract

Deep learning has contributed remarkably to the advancement of time series analysis. Still, deep models can encounter performance bottlenecks in real-world data-scarce scenarios, which can be concealed due to the performance saturation with small models on current benchmarks. Meanwhile, large models have demonstrated great powers in these scenarios through large-scale pre-training. Continuous progress has been achieved with the emergence of large language models, exhibiting unprecedented abilities such as few-shot generalization, scalability, and task generality, which are however absent in small deep models. To change the status quo of training scenario-specific small models from scratch, this paper aims at the early development of *large time series models* (LTSM). During pre-training, we curate large-scale datasets with up to 1 billion time points, unify heterogeneous time series into *single-series sequence* (S3) format, and develop the GPT-style architecture toward LTSMs. To meet diverse application needs, we convert forecasting, imputation, and anomaly detection of time series into a unified *generative task*. The outcome of this study is a Time Series Transformer (Timer), which is generative pre-trained by next token prediction and adapted to various downstream tasks with promising capabilities as an LTSM. Code and datasets are available at: https://github.com/thuml/Large-Time-Series-Model.

1. Introduction

Time series analysis encompasses a broad range of critical tasks, including time series forecasting (Box et al., 2015), imputation (Friedman, 1962), anomaly detection (Breunig et al., 2000), etc. Despite the ubiquity of real-world time series, training samples can be scarce in specific applications. While remarkable advances have been made in deep time series models (Wu et al., 2022; Zeng et al., 2023; Liu et al., 2023b), the accuracy of state-of-the-art deep models (Nie et al., 2022) can still deteriorate drastically in such scenarios, even within prevalent benchmarks as shown in Figure 1. Concurrently, we are witnessing rapid progress of large language models (Radford et al., 2018), involving training on large-scale text corpora and exhibiting remarkable few-shot and zero-shot capabilities (Radford et al., 2019). It can be indicative for the community to develop large time series models (LTSM) that are transferable on various data-scarce scenarios by pre-training on numerous time series data.

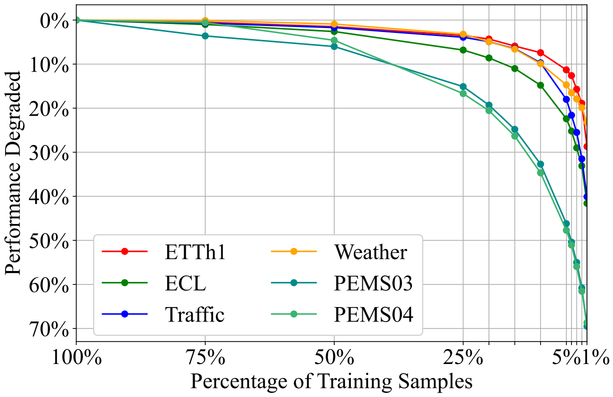

Figure 1. Performance of PatchTST (2022) on different data scarcities. The degradation is reported as the relative increase in MSE compared with training on full samples.

Further, large models evolved by generative pre-training (GPT) have demonstrated several advanced capabilities that are absent in small models: the generalization ability that one model fits many domains, the versatility that one model copes with various scenarios and tasks, and the scalability that performance improves with the scale of parameters and pre-training corpora. Fascinating capabilities have fostered the advancement of artificial general intelligence (OpenAI, 2023). Time series holds comparable practical value to natural language. Essentially, they exhibit inherent similarities in generative modeling (Bengio et al., 2000) and autoregression (Box, 2013). Consequently, the unprecedented success of the generative pre-trained large language models (Zhao et al., 2023) serves as a blueprint for the progress of LTSMs.

[*] Equal contribution [1] School of Software, BNRist, Tsinghua University. Yong Liu <liuyong21@mails.tsinghua.edu.cn>. Haoran Zhang <z-hr20@mails.tsinghua.edu.cn>. Chenyu Li <lichenyu20@mails.tsinghua.edu.cn>. Correspondence to: Mingsheng Long <mingsheng@tsinghua.edu.cn>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

Although unsupervised pre-training on time series data has been widely explored, yielding breakthroughs based on the masked modeling (Zerveas et al., 2021) and contrastive learning (Woo et al., 2022), there are still unsolved fundamental issues for developing LTSMs. Firstly, the dataset infrastructure and unified treatment for heterogeneous time series are lagging behind other fields. As a result, prior unsupervised pre-training methods are typically constrained to a small scale and primarily focus on in-dataset transfer (Zhang et al., 2022; Nie et al., 2022). Secondly, the architecture of scalable large models remains underexplored in the field of time series. It is observed that non-autoregressive structures, which are prevalent and effective in small time series models, may not be suitable for LTSMs. Thirdly, existing large-scale pre-trained models (Woo et al., 2023; Das et al., 2023b) primarily concentrated on a single task (e.g., forecasting), and have scarcely addressed task unification. Consequently, the applicability of LTSMs remains elevatable.

In this paper, we dive into the pre-training and adaptation of large time series models. By aggregating publicly available time series datasets and following curated data processing, we construct *Unified Time Series Dataset (UTSD)* of hierarchical capacities to facilitate the research on the scalability of LTSMs. To pre-train large models on heterogeneous time series data, we propose the *single-series sequence (S3)* format that converts multivariate series with reserved patterns into unified token sequences. For better generalization and versatility, we adopt the GPT-style objective that predicts the next token (Bengio et al., 2000). Eventually, we present **Timer**, a large-scale pre-trained **Time** Series Transfor**mer**. Unlike prevalent encoder-only architecture (Nie et al., 2022; Wu et al., 2022; Das et al., 2023a), Timer exhibits similar characteristics as large language models such as flexible context length and autoregressive generation. It also presents notable few-shot generalization, scalability, and task generality, outperforming state-of-the-art task-specific models on forecasting, imputation, and anomaly detection. Overall, our contributions can be summarized as follows:

- We delve into the LTSM development by curating large-scale datasets comprised of 1B time points, proposing a unified sequence format to cope with data heterogeneity, and presenting Timer, a generative pre-trained Transformer for general time series analysis.
- We apply Timer on various tasks, which is realized in our unified generative approach. Timer exhibits notable feasibility and generalization in each task, achieving state-of-the-art performance with few samples.
- By pre-training on increasing available time series data, Timer exhibits zero-shot forecasting capability. Quantitative evaluations and quality assessments are provided among concurrent large time series models.

2. Related Work

2.1. Unsupervised Pre-training on Sequences

Unsupervised pre-training on large-scale data is the essential step for modality understanding for downstream applications, which has achieved substantial success in sequences, covering natural language (Radford et al., 2021), patch-level image (Bao et al., 2021) and video (Yan et al., 2021). Supported by powerful backbones (Vaswani et al., 2017) for sequential modeling, the paradigms of unsupervised pre-training on sequences have been extensively studied in recent years, which can be categorized into the masked modeling (Devlin et al., 2018), contrastive learning (Chen et al., 2020), and generative pre-training (Radford et al., 2018).

Inspired by significant progress achieved in relevant fields, masked modeling and contrastive learning have been well-developed for time series. TST (Zerveas et al., 2021) and PatchTST (Nie et al., 2022) adopt the BERT-style masked pre-training to reconstruct several time points and patches respectively. LaST (Wang et al., 2022b) proposes to learn the representations of decomposed time series based on variational inference. Contrastive learning is also well incorporated in prior works (Woo et al., 2022; Yue et al., 2022). TF-C (Zhang et al., 2022) constrains the time-frequency consistency by temporal variations and frequency spectrums. SimMTM (Dong et al., 2023) combines masked modeling and contrastive approach within the neighbors of time series.

However, generative pre-training has received relatively less attention in the field of time series despite its prevalence witnessed in developing large language models (Touvron et al., 2023; OpenAI, 2023). Most large language models are generative pre-trained (Zhao et al., 2023) with token-level supervision, where each token is generated based on the previous context and independently supervised (Bengio et al., 2000). Consequently, they are not constrained by specific input and output lengths and excel at multi-step generation. Furthermore, prior studies (Wang et al., 2022a; Dai et al., 2022) have demonstrated that scalability and generalization largely stem from generative pre-training, which requires more training data than other pre-training paradigms. Thus, our work aims to investigate and revitalize generative pre-training towards LTSMs, facilitated by extensive time series and deftly designed adaptation on downstream tasks.

2.2. Large Time Series Models

Pre-trained models with scalability can evolve to large foundation models (Bommasani et al., 2021), featured by increasing model capacity and pre-training scale to solve various data and tasks. Large language models even demonstrate advanced capabilities such as in-context learning and emergent abilities (Wei et al., 2022). As of present, research on large time series models remains at a nascent stage. Ex-

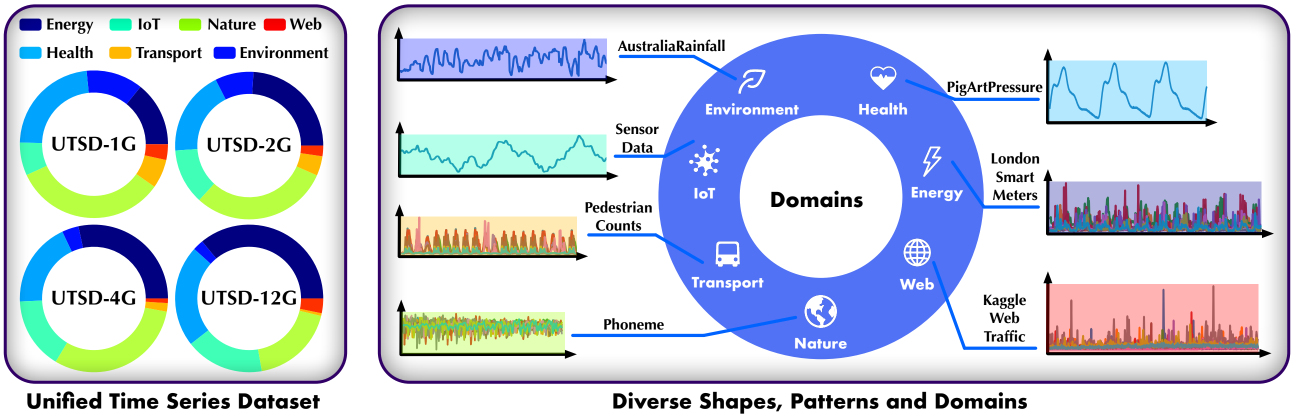

Figure 2. Illustration of Unified Time Series Dataset (UTSD) that is composed of various time series domains with hierarchical capacities.

isting efforts towards LTSMs can be categorized into two groups, with one being large language models for time series. FPT (Zhou et al., 2023) regards GPT-2 as a representation extractor of sequences, which is respectively fine-tuned on different downstream tasks. LLMTime (Chang et al., 2023) encodes time series into numerical tokens for LLMs, exhibiting model scalability in the forecasting task. Time-LLM (Jin et al., 2023) investigates prompting techniques to enhance prediction, demonstrating the generalization ability of LLMs. Unlike these methods, Timer is pre-trained natively on time series and free from extra modality alignment.

Another category includes pre-trained models on large-scale time series. ForecastFPN (Dooley et al., 2023) is trained on synthetic series for zero-shot forecasting. CloudOps (Woo et al., 2023) adopts masked modeling on Transformer for domain-specific forecaster. Lag-Llama (Rasul et al., 2023) is a probabilistic univariate forecaster that adopts lags as covariates. PreDcT (Das et al., 2023b) is a decoder-only Transformer pre-trained on Google Trends, exhibiting notable zero-shot ability. TimeGPT-1 (Garza & Mergenthaler-Canseco, 2023) releases the first commercial API for zero-shot forecasting. Different from prior works, our UTSD contains 1B real-world time points, which is not a simple aggregation but follows curated data processing. Timer is applicable to downstream tasks beyond forecasting and exhibits promising scalability. We are also the first to establish a zero-shot forecasting benchmark on concurrent LTSMs.

3. Approach

Inspired by the sequential structure inherent in language and time series, we leverage the advancement of large language models for developing LTSMs. In this paper, we advocate the development of large models for time series with (1) the utilization of extensive time series corpora, (2) the adoption of a standardized format for diverse time series data, and (3) the generative pre-training on the decoder-only Transformer that autoregressively predict the next time series token.

3.1. Data

Large-scale datasets are of paramount importance for pre-training large models. However, the curation of time series datasets can be prohibitively challenging. In spite of their ubiquity, there are numerous data of low quality, including missing values, unpredictability, variance in shape, and irregular frequencies, which significantly impact the efficacy of pre-training. Therefore, we establish the criteria for filtering high-quality data and stacking up the hierarchy of time series corpora. Concretely, we record the statistics of each dataset, including (1) basic properties, such as time steps, variate number, file size, frequency, etc; and (2) time series characteristics: periodicity, stationarity, and predictability. This also allows us to assess the complexity of different datasets and progressively conduct scalable pre-training.

We curate Unified Time Series Dataset (UTSD) as shown in Figure 2. UTSD is constructed with hierarchical capacities to facilitate the scalability research of large models. UTSD encompasses seven domains with up to 1 billion time points (UTSD-12G), covering typical scenarios of time series analysis. Following the principle of keeping pattern diversity, we include as diverse datasets as possible in each hierarchy, ensure the data size of each domain is nearly balanced when scaling up, and the complexity gradually increases in accordance with the calculated statistics. We release four volumes on https://huggingface.co/datasets/thuml/UTSD.

Notably, we make our curation applicable to the increasing open-source datasets, which is beneficial for the continuous expansion of time series corpora. Particularly, we conduct the same procedure on the recent LOTSA (Woo et al., 2024), a great endeavor with 27B time points, to explore zero-shot forecasting and establish the benchmark of LTSMs. Detailed construction and statistics are provided in Appendix A.

3.2. Training Strategy

Different from natural language, which has been facilitated by the well-established discrete tokenization and sequential structure with the regular shape, constructing unified time series sequences is not straightforward due to the heterogeneity of series such as amplitude, frequency, stationarity, and disparities of the datasets in the variate number, series length and purpose. To promote pre-training on extensive time series, we propose to convert heterogeneous time series into *single-series sequence (S3)*, which reserves the patterns of series variations with the unified context length.

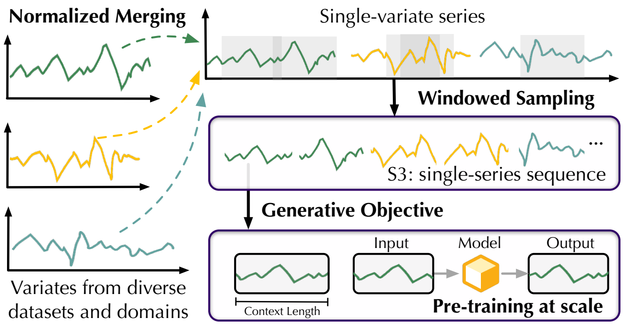

Figure 3. Pre-training strategy for heterogeneous time series.

As depicted in Figure 3, our initial step involves normalizing and merging at the level of variates. Each series representing a variate will be divided into training and validation splits at a ratio of 9:1 for pre-training. We apply the statistics of the training split to normalize the entire series. The normalized time series are merged into a pool of single-variate series. The time points of single-variate series for training follow the normal distribution, which mitigates the discrepancies in the amplitude and variate numbers across multiple datasets.

We uniformly sample sequences from the pool by a window, obtaining single-series sequences with a fixed context length, as the format of S3. The proposed format is essentially an extension of Channel Independence CI (Nie et al., 2022). However, CI necessitates time-aligned multivariate series and flattens the variate dimension to the same batch, thereby requiring the batch of samples to originate from the same dataset. Based on our format, the model observes sequences from different periods and different datasets, thus increasing the pre-training difficulty and directing more attention to the temporal variation. S3 does not require time alignment, which applies to broad univariate and irregular time series. We then employ generative pre-training, where single-series sequences are regarded as standard sentences of time series.

3.3. Model Design

Given the limited exploration of the backbone for large time series models, we extensively evaluate candidate backbones on the same pre-training scale in Section 4.5, which validates Transformer as the scalable choice. Further, we review Transformer-based models in time series forecasting, which have experienced notable development in recent years. They can be categorized into encoder-only and decoder-only architectures following a similar pipeline. As illustrated in Figure 4, prevalent small time series forecasters, the encoder-only non-autoregressive models, generate predictions with the globally flattened representation of lookback series. Although direct projection may benefit from end-to-end supervision, flattening can also wipe out sequential dependencies modeled by attention, thereby weakening Transformer layers to reveal the patterns of temporal variations.

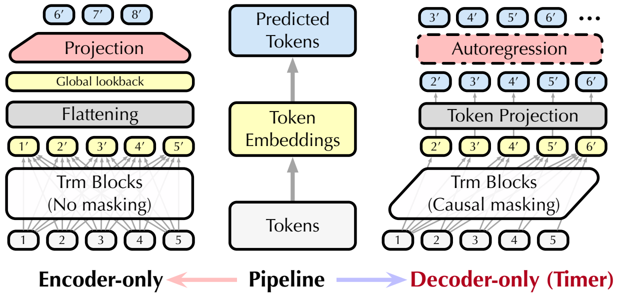

Figure 4. Architectures of typical Transformer-based forecasters.

Inspired by the substantial progress of decode-only LLMs with the ability for iterative generation, we opt for an under-explored autoregressive approach for generative pre-training. As language models autoregressively predict the next token:

$$P(\mathcal{U}) = \prod_{i=1}^{N} p(u_i | u_{<i}) \tag{1}$$

on the token sequence $\mathcal{U} = \{u_1, \ldots, u_N\}$, we first establish the tokenization of the given single-series sequence (S3) $\mathbf{X} = \{x_1, \ldots, x_{NS}\}$ with the unified context length NS. We define the time series token as consecutive time points (segment) of length S that encompass the series variations:

$$\mathbf{s}_i = \{x_{(i-1)S+1}, \ldots, x_{iS}\} \in \mathbb{R}^S, \ i = 1, \ldots, N. \tag{2}$$

We adopt the decoder-only Transformer with dimension D and L layers and apply generative pre-training (GPT) on N tokens in the single-series sequence (sentence):

$$\begin{aligned} \mathbf{h}_i^0 &= \mathbf{W}_e \mathbf{s}_i + \mathbf{TE}_i, \ i = 1, \ldots, N, \\ \mathbf{H}^l &= \mathrm{TrmBlock}(\mathbf{H}^{l-1}), \ l = 1, \ldots, L, \\ \{\hat{\mathbf{s}}_{i+1}\} &= \mathbf{H}^L \mathbf{W}_d, \ i = 1, \ldots, N, \end{aligned} \tag{3}$$

where $\mathbf{W}_e, \mathbf{W}_d \in \mathbb{R}^{D\times S}$ encode and decode token embeddings in $\mathbf{H} = \{\mathbf{h}_i\} \in \mathbb{R}^{N\times D}$ independetly, and $\mathbf{TE}_i$ denotes the optional timestamp embedding. Via the causal attention of the decoder-only Transformer, the autoregressively generated $\hat{\mathbf{s}}_{i+1}$ is obtained as the next token of $\mathbf{s}_i$. Thus, we formulate the pre-training objective as follows:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{NS} \sum ||\mathbf{s}_i - \hat{\mathbf{s}}_i||_2^2, \ i = 2, ..., N. \tag{4}$$

Equation 4 yields token-wise supervising signals, where generated tokens of each position are independently supervised.

Consequently, generative pre-trained models are endowed with the flexibility to address unfixed context length during inference and excel at multi-step generation by iteratively sliding and enlarging input tokens. While small time series models generally refrain from iterative multi-step prediction to mitigate error accumulations, our experiments reveal that autoregressive models pre-trained on large-scale data can also perform as competitively as direct multi-step predictors.

4. Experiments

We demonstrate Timer as a large time series model in time series forecasting, imputation, and anomaly detection by tackling them in a unified generative scheme, which is described in Figure 5. We compare Timer with state-of-the-art task-specific models and present the benefit of pre-training on data-scarce scenarios, known as the few-shot ability of large models. Furthermore, we delve into the model scalability, including model/data size, and try to build a comprehensive zero-shot evaluation across concurrent large time series models. All the downstream datasets are not included in the pre-training stage to prevent data leakage. We provide the detailed implementation and model configurations of pre-training and adaptation in Appendix B.1 and B.2.

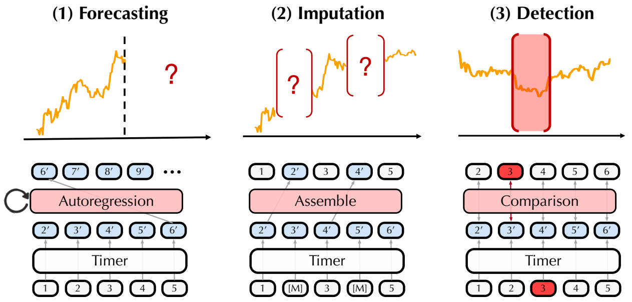

Figure 5. Illustration of our generative task unification: (1) Generative pre-trained Timer can naturally predict the next series by the iterative autoregression; (2) By introducing masked tokens during adaptation, Timer generates imputations with the previous context and assemble them with the observed part; (3) We propose predictive anomaly detection by predicting normal series in advance.

4.1. Time Series Forecasting

Setups Time series forecasting is essential and presents challenges in real-world applications. To thoroughly evaluate the performance, we elaborately establish the benchmark, including ETT, ECL, Traffic, Weather, and PEMS adopted in Liu et al. (2023b). We adopt the unified lookback length as 672 and the forecast length as 96. We pre-training Timer on UTSD-12G with the segment length $S = 96$ and the number of tokens $N = 15$, such that Timer can deal with time series with a context length up to 1440. The downstream forecasting task can be naturally completed as the next token prediction, which is detailed in Appendix B.2.

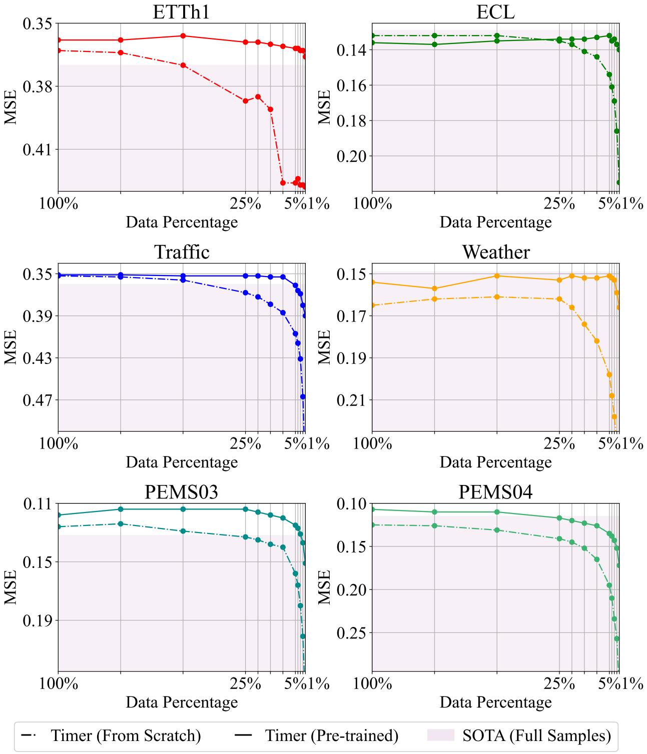

Figure 6. Forecasting performance of Timer obtained by training from scratch and fine-tuning from the pre-trained model on different data scarcities. State-of-the-art small deep forecasters trained on full samples are provided the SOTA baseline. A smaller MSE indicates better results. Detailed results are provided in Table 10.

Results As depicted in Figure 6, we present the results of the pre-trained Timer (solid line) and Timer trained from scratch (dashed line) under different data scarcities. We also evaluate state-of-the-art forecasters by training them on full samples as a competitive baseline. Concretely, we train PatchTST (Nie et al., 2022) and iTransformer (Liu et al., 2023b) on each dataset and report the better one as SOTA. Timer fine-tuned on few training samples demonstrates competitive results as advanced small deep forecasters, specifically achieving better results with only 1% available samples from ETTh1, 5% from Traffic, 3% from PEMS03, and 25% from PEMS04 and exhibiting remarkable few-shot ability.

To assess pre-training benefit, we compare solid and dashed lines, differing by whether to load the pre-trained checkpoint. Concretely, the performance of training a random-initialized Timer on full samples can be achieved by fine-tuning our pre-trained Timer with only 2% of the training samples in ETTh1, 5% in ECL, 1% in Weather, and 4% in PEMS03, which exemplifies the transferable knowledge acquired by pre-training on UTSD. When all samples are available, the performance of the pre-trained Timer can also outperform training it from scratch: the prediction error is reduced as $0.165 \to 0.154$ on Weather, $0.126 \to 0.118$ on PEMS03, and $0.125 \to 0.107$ on PEMS04. Overall, in widespread

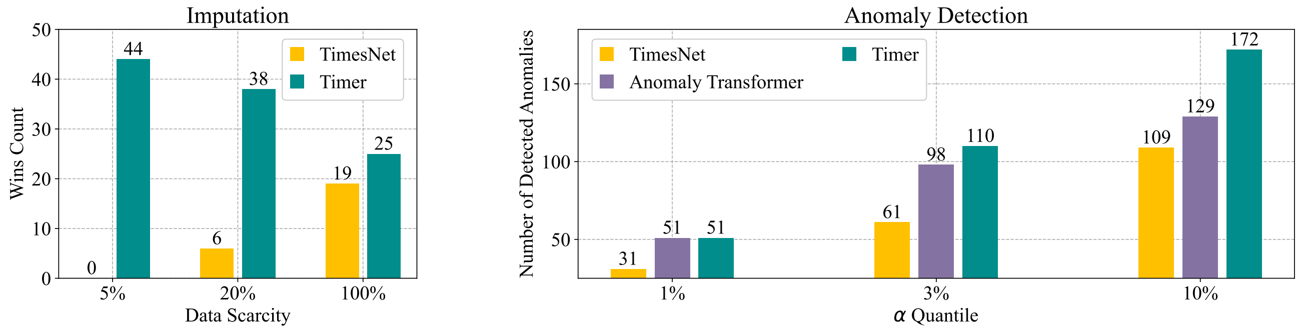

Figure 7. Performance comparison with state-of-the-art small deep models. For imputation, Time is compared with TimesNet (Wu et al., 2022) under different data scarcities, each of which contains 44 imputation scenarios. In UCR Anomaly Detection Archive (Wu & Keogh, 2021), we compare the number of detected anomalies under given confidence quantiles. Detailed results are provided in Table 11-15.

data-scarce scenarios, the performance degradation can be alleviated by the few-shot generalization of LTSMs.

4.2. Imputation

Setups Imputation is ubiquitous in real-world applications, aiming to fill corrupted time series based on partially observed data. However, while various machine learning algorithms and simple linear interpolation can effectively cope with the corruptions randomly happening at the point level, real-world corruptions typically result from prolonged monitor shutdowns and require a continuous period of recovery. Consequently, imputation can be ever challenging when attempting to recover a span of time points encompassing intricate series variations. In this task, we conduct the segment-level imputation. Each time series is divided into 8 segments and each segment has the length of 24 and the possibility of being completely masked. We obtain Timer on UTSD-4G by generative pre-training with the segment length $S = 24$ and the token number $N = 15$. For downstream adaptation, we conduct the denoising autoencoding in T5 (Raffel et al., 2020) as detailed in Appendix B.2 to recover the masked spans in a generative way.

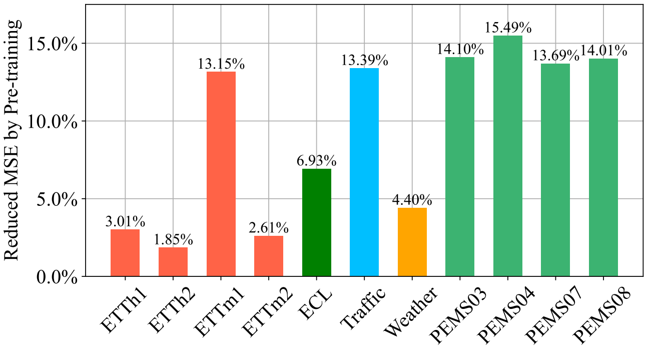

Figure 8. Pre-training benefit of Timer on the downstream imputation task with 5% available samples. Following TimesNet (Wu et al., 2022), each dataset is imputed with four mask ratios in $\{12.5\%, 25\%, 37.5\%, 50\%\}$ and we calculate the average reduced imputation error in MSE relative to training from scratch. Additional results of other data scarcities are provided in Figure 18.

Results We establish a comprehensive segment-level imputation benchmark, which includes 11 datasets with four mask ratios each. Timer is compared with the previous state-of-the-art imputation model (Wu et al., 2022). As shown in the left of Figure 7, Timer outperforms in respectively 100.0%, 86.4%, and 56.8% of 44 imputation scenarios under the data scarcities of $\{5\%, 20\%, 100\%\}$, validating the effectiveness of Timer on the challenging imputation task. Regarding the benefit of pre-training, we present the promotion as the reduction ratio of imputation errors in Figure 8, where pre-training consistently yields positive effects with 5% downstream samples. Additional experiments on 20% and 100% available samples are provided in Figure 18, which still present notable performance improvement.

4.3. Anomaly Detection

Setups Anomaly detection is vital in industry and operations. Previous methods (Xu et al., 2021; Wu et al., 2022) typically tackle the unsupervised scenario in a reconstructive approach, where a model is trained to reconstruct the input series, and the output is regarded as the normal series. Based on our generative model, we cope with anomaly detection in a predictive approach, which utilizes the observed segments to predict the future segment, and the predicted segment will be established as the standard to be compared with the actual value received. Unlike the previous method requiring to collect time series of a period for reconstruction, our predictive approach allows for segment-level anomaly detection on the fly. Thus, the task is converted into a next token prediction task as detailed in Appendix B.2.

We introduce UCR Anomaly Archive (Wu & Keogh, 2021) that contains 250 tasks. In each task, a single normal time series is provided for training, and the model should locate the position of an anomaly in the test series. We first train a predictive model on the training set and calculate the MSE between the predicted series and ground truth on the test set. By regarding the MSE of all segments as the confidence level, the segments with higher than α quantile of confidence are labeled as potential positions of anomalies.

Results We evaluate well-acknowledged anomaly detection models, including TimesNet (Wu et al., 2022) and Anomaly Transformer (Xu et al., 2021). As shown in the right of Figure 7, we present the number of detected anomalies with given quantiles, where Timer outperforms other advanced anomaly detection models, exhibiting the versatility of our generative time series model. Moreover, Figure 9 compares the detection performance of pre-trained models and from scratch using two indicators. The left figure shows the number of datasets that the model has completed detection within the quantile of 3% and 10%, and the right figure shows the quantile distribution and the averaged quantile of all 250 UCR datasets, where the pre-trained Timer with the smaller averaged quantile works as a more accurate detector.

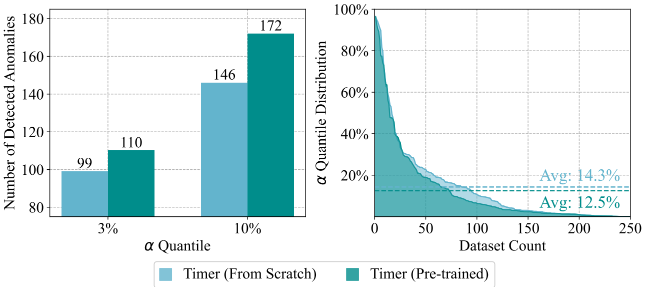

Figure 9. Downstream anomaly detection results of Timer obtained by training from scratch and adapting with the pre-trained model.

4.4. Scalability

Scalability is the essential property that emerges from pre-trained models to large models. To investigate the scaling behavior of Timer, we pre-train Timer with increased model size and data size as detailed in Appendix B.1 and evaluate it in downstream forecasting on all subsets of PEMS.

Model size We keep UTSD-4G as the pre-training set. Results are presented in Figure 10. While keeping model dimension $D = 256$, we increase the number of layers. The growth of parameters from 1M to 4M leads to the decrease in forecasting errors in two few-shot scenarios by an average of 14.7% and 20.6% respectively. Subsequently, we increase the model dimension under the fixed layer number $L = 6$, enlarging parameters from 3M to 50M, resulting in further improved performance of 25.1% and 18.2%, validating the efficacy of scaling up the model size.

Data scale We pre-train Timer under the same model size with different UTSD sizes, which exhibits steady improvement with the enlarged pre-training scale in Figure 11. The benefit is relatively small compared to expanding the model size previously, which can be attributed to the performance saturation on these datasets. Compared with large language models, the parameter scale of Timer can still be small, indicating the higher parameter efficiency in the time series modality, which is also supported by prior works (Das et al., 2023b). As the scaling law (Kaplan et al., 2020) of large models highlights the significance of synchronized scaling of data with the model parameters, there is still an urgent need to accelerate the data infrastructure in the time series field to promote the development of LTSMs.

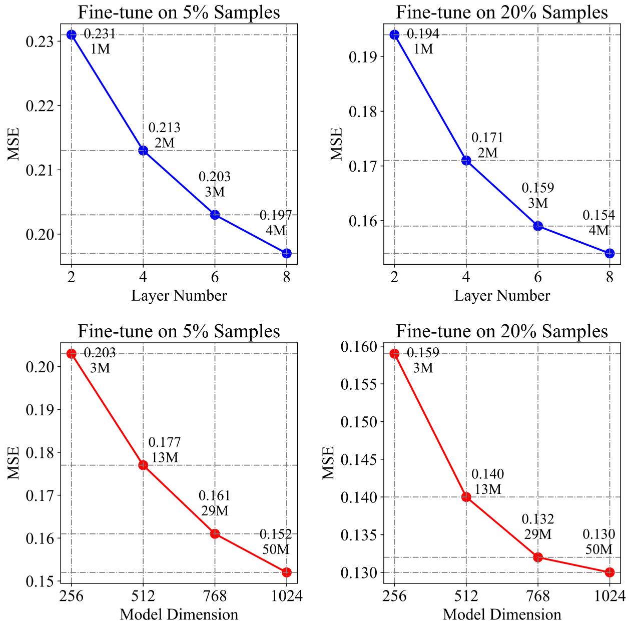

Figure 10. Larger Timer demonstrates better performance on downstream forecasting. Models are all pre-trained on UTSD-4G. Detailed results of all PEMS subsets are provided in Table 16.

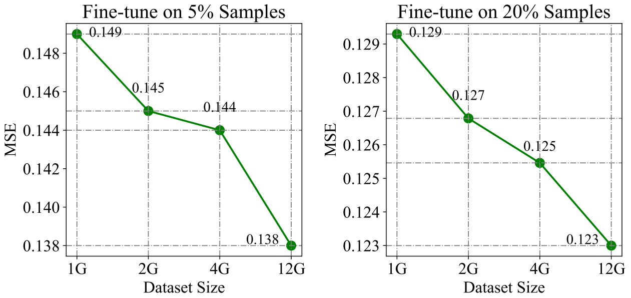

Figure 11. Timer trained on larger dataset demonstrates better performance on downstream forecasting. Models are configured with $L = 8$ and $D = 1024$. Detailed results are provided in Table 16.

Overall, by increasing the model size and data scale, Timer reduces the prediction error as $0.231 \to 0.138$ (-40.3%) and $0.194 \to 0.123$ (-36.6%) under few-shot scenarios, surpassing state-of-the-art multivariate forecaster (Liu et al., 2023b) training on full samples of PEMS datasets (0.139).

4.5. Model Analysis

Backbone for LTSM Deep learning approaches have brought the boom of time series analysis, with various backbones for modeling the sequential time series modality being proposed. To validate the appropriate option for large

Table 1. Downstream forecasting results under different data scarcity of the encoder-only and decoder-only Transformer respectively pre-trained on UTST-12G. Datasets are ordered by the degradation in Figure 1. Full results of PEMS and ETT can be found in Table 17.

SCENARIO	1% TARGET				5% TARGET				20% TARGET			
ARCHITECTURE	ENCODER		DECODER		ENCODER		DECODER		ENCODER		DECODER	
PRE-TRAINED	NONE	12G	NONE	12G	NONE	12G	NONE	12G	NONE	12G	NONE	12G
PEMS (AVG)	0.286	0.246	0.328	**0.180**	0.220	0.197	0.215	**0.138**	0.173	0.164	0.153	**0.126**
ECL	0.183	0.168	0.215	**0.140**	0.150	0.147	0.154	**0.132**	0.140	0.138	0.137	**0.134**
TRAFFIC	0.442	0.434	0.545	**0.390**	0.392	0.384	0.407	**0.361**	0.367	0.363	0.372	**0.352**
ETT (AVG)	0.367	0.317	0.340	**0.295**	0.339	0.303	0.321	**0.285**	0.309	0.301	0.297	**0.288**
WEATHER	0.224	0.165	0.246	**0.166**	0.182	0.154	0.198	**0.151**	0.153	**0.149**	0.166	0.151

time series models, we compare Timer with four candidates: MLP-based TiDE (Das et al., 2023a), CNN-based TCN (Bai et al., 2018), RNN-based LSTM (Hochreiter & Schmidhuber, 1997) and encoder-only PatchTST (Nie et al., 2022).

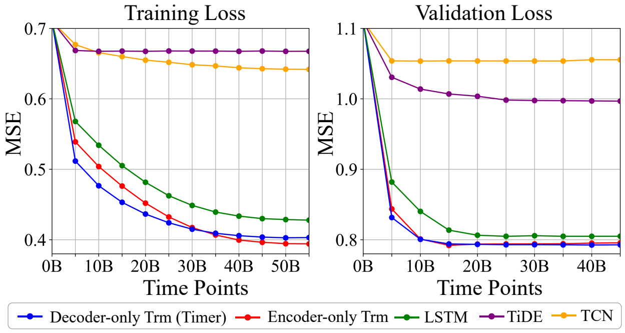

Figure 12. Training loss of candidate backbones. Model dimension and layer number are consistently chosen for a fair comparison.

To make sure the evaluation is comparable on different backbones, we maintain the same model configuration, including the model dimension and layer number, and pre-train these backbones on our UTSD-4G respectively. We set the token length $S = 96$ and the context length as 672 for Timer. For other non-autoregressive backbones, we pre-train them by direct multi-step forecasting in the 672-pred-96 setting. The loss curves of training and validation are calculated as the MSE of the same set of model outputs (length-96 time series). As illustrated in Figure 12, Transformer exhibits excellent scalable ability as the backbone for LTSMs, whereas MLP-based and CNN-based architectures may encounter the bottleneck in accommodating diverse time series data.

Decoder-only v.s. Encoder-only While a smaller training loss is achieved by the encoder-only Transformer in Figure 12, the progress of large language models indicates that decoder-only models may possess stronger generalization capabilities in downstream adaptation (Wang et al., 2022a; Dai et al., 2022), which is the essential purpose of LTSMs. Therefore, we proceed to compare their forecasting performance under varying degrees of data scarcity.

We elaborately evaluate two architectures on six benchmarks in Table 1. In the case of training from scratch (Pre-trained = None), the encoder-only Transformer will achieve better performance if the training samples are insufficient (Target = 1%). Instead, the decoder-only architecture will demonstrate improved performance when more training samples are provided in the end-to-end scenarios. After pre-training on UTSD-12G (Pre-trained = 12G), Timer as the decoder-only Transformer achieves the best performance in most downstream scenarios, indicating better generalization than the encoder-only pre-trained model. The observations are consistent with several findings in large language models and elucidate why the encoder-only structure has become prevalent in the field of time series. Existing benchmarks can still be small and the encoder-only model can overfit in end-to-end scenarios. Meanwhile, the decoder-only Transformer, which excels at generalizing on different domains, is a promising choice for developing large time series models.

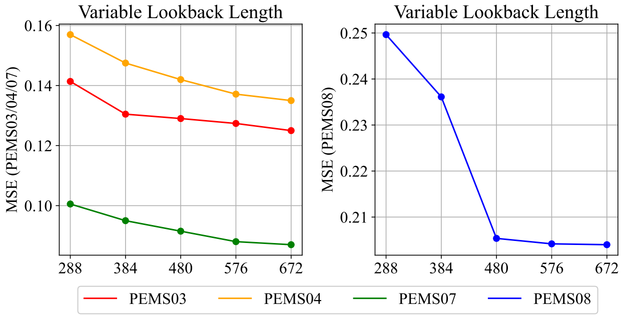

Figure 13. Performance of one Timer for all lookback lengths.

Flexible sequence length Typically, current deep forecasting models are trained on specific lookback and forecast lengths, limiting their versatility. Instead, the decoder-only architecture can provide the flexibility to address different sequence lengths. For instance, one Timer is applicable on different lookback lengths because of token-wise supervision outlined in Equation 4. In addition to the feasibility, it achieves enhanced performance by increasing the lookback length in Figure 13. As for the forecast length, increasing works (Liu et al., 2024) bring the revival of autoregression

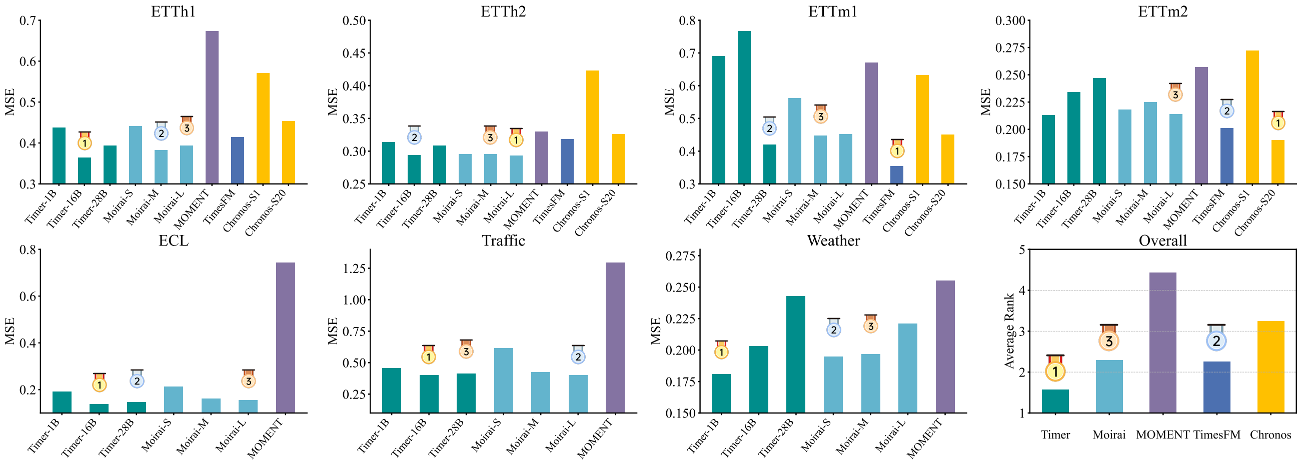

Figure 14. Zero-shot evaluation on LTSMs. The top three models for each dataset are highlighted on the leaderboard. *Average Rank* of each model is calculated on the benchmarks in which the model has participated. Detailed results are provided in Table 18.

(iterative multi-step prediction), enabling the generation of future predictions with arbitrary lengths. We explore this paradigm by rolling one model for all forecast lengths in Figure 15, where the decoder-only Times exhibits smaller error accumulation, thereby achieving better performance.

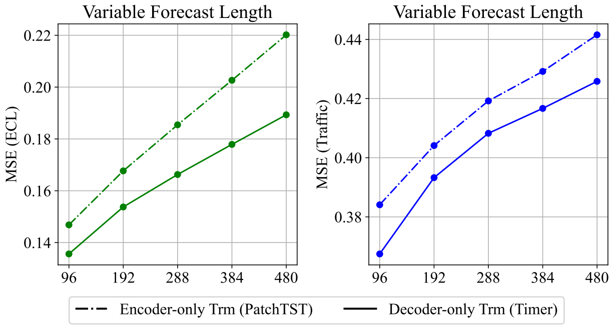

Figure 15. Performance of Timer/PatchTST for all forecast lengths. We conduct rolling forecasting on a single 672-pred-96 model.

4.6. Evaluation of Large Time Series Models

There is a growing surge in the development of large models in the field of time series (Garza & Mergenthaler-Canseco, 2023; Das et al., 2023b; Woo et al., 2024; Ansari et al., 2024; Goswami et al., 2024). One particularly fascinating direction of research is focused on zero-shot forecasting (ZSF), which has the potential to renovate the conventional practice of training small models or fine-tuning language models for each specific scenario. Zero-shot generalization represents a sophisticated capability of large models, necessitating substantial model capacity and pre-training on extensive datasets. Consequently, we are actively expanding our dataset by incorporating the latest data infrastructure (Woo et al., 2024) in this field to pre-train Timer on ever larger scales (1B/16B/28B). Given the significant value to researchers and practitioners, we extensively evaluate concurrent large models and establish the first zero-shot forecasting benchmark of LTSMs as detailed in Appendix B.2.

Quality assessments Our evaluation assesses the quality of LTSMs in Table 9, including (1) fundamental attributes such as pre-training scale, parameters; (2) abilities such as applicable tasks, context length, etc. Current LTSMs essentially build upon Transformer, with a significantly smaller number of parameters compared to LLMs. There is still potential to support more tasks and longer contexts.

Quantitative evaluations We apply official checkpoints on seven datasets that do not appear during pre-training. The performance is fairly evaluated using MSE by predicting future 96 points of all windows in each dataset. Figure 14 presents the result and rank of each model, where the top-ranked LTSMs are Timer, Moiria (Woo et al., 2024), and TimesFM (Das et al., 2023b). However, the positive correlation between performance and pre-training scale remains relatively weak, highlighting the significance of high-quality data and synchronized scaling of data and model size.

5. Conclusion and Future Work

Real-world time series analysis is increasingly underscoring the demand for large time series models (LTSM). In this paper, we release a time series dataset with 1 billion time points, propose a unified sequence format to address the heterogeneity of multivariate time series, and develop a generative pre-trained Transformer as a generalizable, scalable, task-general LTSM. Empirically, we evaluate our model in forecasting, imputation, and anomaly detection, yielding state-of-the-art performance and notable pre-training benefits in the data-scarce scenario. Further analysis validates the model scalability, explores the architecture for LTSMs, and highlights the versatility of our autoregressive generation. By performing zero-shot forecasting on available large models, we conduct the initial quantitative assessments among LTSMs. Quality evaluations unveil crucial pathways for future development, including better zero-shot generalization and facilitating probabilistic and long-context forecasting.

Impact Statement

This paper aims to advance the development of large models for time series. In this work, we release a high-quality and unified time series dataset for scalable pre-training, which can serve as a foundation for pre-training and establishing new benchmarks. The outcome large model demonstrates notable effectiveness of generalization, versatility across various tasks, and scalability to refine performance, offering valuable insights for future investigations and application values for practitioners. Our paper mainly focuses on scientific research and has no obvious negative social impact.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (62022050 and U2342217), the BNRist Innovation Fund (BNR2024RC01010), and the National Engineering Research Center for Big Data Software.

References

Alexandrov, A., Benidis, K., Bohlke-Schneider, M., Flunkert, V., Gasthaus, J., Januschowski, T., Maddix, D. C., Rangapuram, S., Salinas, D., Schulz, J., et al. Gluonts: Probabilistic and neural time series modeling in python. *Journal of Machine Learning Research*, 21(116): 1–6, 2020.

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Arango, S. P., Kapoor, S., et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Bai, S., Kolter, J. Z., and Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Bao, H., Dong, L., Piao, S., and Wei, F. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Bengio, Y., Ducharme, R., and Vincent, P. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.

Bergmeir, C., Bui, Q., de Nijs, F., and Stuckey, P. Residential Power and Battery Data, 2023. URL https://doi.org/10.5281/zenodo.8219786.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Box, G. Box and jenkins: time series analysis, forecasting and control. In *A Very British Affair: Six Britons and the Development of Time Series Analysis During the 20th Century*, pp. 161–215. Springer, 2013.

Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 93–104, 2000.

CDC. Flu portal dashboard, 2017. URL https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html. Accessed: [insert date of access].

Chang, C., Peng, W.-C., and Chen, T.-F. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.

Chen, S. Beijing Multi-Site Air Quality. UCI Machine Learning Repository, 2019. DOI: https://doi.org/10.24432/C5RK5G.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Dai, D., Sun, Y., Dong, L., Hao, Y., Sui, Z., and Wei, F. Why can gpt learn in-context? language models secretly perform gradient descent as meta optimizers. *arXiv preprint arXiv:2212.10559*, 2022.

Das, A., Kong, W., Leach, A., Sen, R., and Yu, R. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023a.

Das, A., Kong, W., Sen, R., and Zhou, Y. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*, 2023b.

Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., and Keogh, E. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dong, J., Wu, H., Zhang, H., Zhang, L., Wang, J., and Long, M. Simmtm: A simple pre-training framework for masked time-series modeling. *arXiv preprint arXiv:2302.00861*, 2023.

Dooley, S., Khurana, G. S., Mohapatra, C., Naidu, S., and White, C. Forecastpfn: Synthetically-trained zero-shot forecasting. *arXiv preprint arXiv:2311.01933*, 2023.

Elliott, G., Rothenberg, T. J., and Stock, J. H. Efficient tests for an autoregressive unit root. *Econometrica*, 1996.

Emami, P., Sahu, A., and Graf, P. Buildingsbench: A large-scale dataset of 900k buildings and benchmark for short-term load forecasting. *Advances in Neural Information Processing Systems*, 2023.

Friedman, M. The interpolation of time series by related series. *Journal of the American Statistical Association*, 57(300):729–757, 1962.

Garza, A. and Mergenthaler-Canseco, M. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

Godahewa, R., Bergmeir, C., Webb, G. I., Hyndman, R. J., and Montero-Manso, P. Monash time series forecasting archive. *arXiv preprint arXiv:2105.06643*, 2021.

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

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

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Jiang, J., Han, C., Jiang, W., Zhao, W. X., and Wang, J. Libcity: A unified library towards efficient and comprehensive urban spatial-temporal prediction. *arXiv preprint arXiv:2304.14343*, 2023.

Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J. Y., Shi, X., Chen, P.-Y., Liang, Y., Li, Y.-F., Pan, S., et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015. URL http://arxiv.org/abs/1412.6980.

Liu, M., Zeng, A., Chen, M., Xu, Z., Lai, Q., Ma, L., and Xu, Q. Scinet: time series modeling and forecasting with sample convolution and interaction. *NeurIPS*, 2022.

Liu, X., Xia, Y., Liang, Y., Hu, J., Wang, Y., Bai, L., Huang, C., Liu, Z., Hooi, B., and Zimmermann, R. Largest: A benchmark dataset for large-scale traffic forecasting. In *Advances in Neural Information Processing Systems*, 2023a.

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

Liu, Y., Qin, G., Huang, X., Wang, J., and Long, M. Autotimes: Autoregressive time series forecasters via large language models. *arXiv preprint arXiv:2402.02370*, 2024.

Mancuso, P., Piccialli, V., and Sudoso, A. M. A machine learning approach for forecasting hierarchical time series. *Expert Systems with Applications*, 182:115102, 2021.

Mouatadid, S., Orenstein, P., Flaspohler, G., Oprescu, M., Cohen, J., Wang, F., Knight, S., Geogdzhayeva, M., Levang, S., Fraenkel, E., et al. Subseasonalclimateusa: A dataset for subseasonal forecasting and benchmarking. *Advances in Neural Information Processing Systems*, 36, 2024.

Nguyen, T., Jewik, J., Bansal, H., Sharma, P., and Grover, A. Climatelearn: Benchmarking machine learning for weather and climate modeling. *Advances in Neural Information Processing Systems*, 36, 2024.

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

OpenAI, R. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2:13, 2023.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Rasul, K., Ashok, A., Williams, A. R., Khorasani, A., Adamopoulos, G., Bhagwatkar, R., Biloš, M., Ghonia, H., Hassen, N. V., Schneider, A., et al. Lag-llama: Towards foundation models for time series forecasting. *arXiv preprint arXiv:2310.08278*, 2023.

Tan, C. W., Bergmeir, C., Petitjean, F., and Webb, G. I. Time series extrinsic regression: Predicting numeric values from time series data. *Data Mining and Knowledge Discovery*, 35:1032–1060, 2021.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

van Panhuis, W. G., Cross, A., and Burke, D. S. Project tycho 2.0: a repository to improve the integration and reuse of data for global population health. *Journal of the American Medical Informatics Association*, 25(12): 1608–1617, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, T., Roberts, A., Hesslow, D., Le Scao, T., Chung, H. W., Beltagy, I., Launay, J., and Raffel, C. What language model architecture and pretraining objective works best for zero-shot generalization? In *International Conference on Machine Learning*, pp. 22964–22984. PMLR, 2022a.

Wang, Y., Han, Y., Wang, H., and Zhang, X. Contrast everything: A hierarchical contrastive framework for medical time-series. *arXiv preprint arXiv:2310.14017*, 2023a.

Wang, Z., Xu, X., Zhang, W., Trajcevski, G., Zhong, T., and Zhou, F. Learning latent seasonal-trend representations for time series forecasting. *Advances in Neural Information Processing Systems*, 35:38775–38787, 2022b.

Wang, Z., Wen, Q., Zhang, C., Sun, L., Von Krannichfeldt, L., and Wang, Y. Benchmarks and custom package for electrical load forecasting. *arXiv preprint arXiv:2307.07191*, 2023b.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Woo, G., Liu, C., Sahoo, D., Kumar, A., and Hoi, S. Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. *arXiv preprint arXiv:2202.01575*, 2022.

Woo, G., Liu, C., Kumar, A., and Sahoo, D. Pushing the limits of pre-training for time series forecasting in the cloudops domain. *arXiv preprint arXiv:2310.05063*, 2023.

Woo, G., Liu, C., Kumar, A., Xiong, C., Savarese, S., and Sahoo, D. Unified training of universal time series forecasting transformers. *arXiv preprint arXiv:2402.02592*, 2024.

Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., and Long, M. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

Wu, R. and Keogh, E. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

Xu, J., Wu, H., Wang, J., and Long, M. Anomaly transformer: Time series anomaly detection with association discrepancy. *arXiv preprint arXiv:2110.02642*, 2021.

Yan, W., Zhang, Y., Abbeel, P., and Srinivas, A. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2021.

Yue, Z., Wang, Y., Duan, J., Yang, T., Huang, C., Tong, Y., and Xu, B. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8980–8987, 2022.

Zeng, A., Chen, M., Zhang, L., and Xu, Q. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 2023.

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

Zhang, X., Zhao, Z., Tsiligkaridis, T., and Zitnik, M. Self-supervised contrastive pre-training for time series via time-frequency consistency. *Advances in Neural Information Processing Systems*, 35:3988–4003, 2022.

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

Zheng, Y., Yi, X., Li, M., Li, R., Shan, Z., Chang, E., and Li, T. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 2267–2276, 2015.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. 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.

Zhou, J., Lu, X., Xiao, Y., Su, J., Lyu, J., Ma, Y., and Dou, D. Sdwpf: A dataset for spatial dynamic wind power forecasting challenge at kdd cup 2022. *arXiv preprint arXiv:2208.04360*, 2022.

Zhou, T., Niu, P., Wang, X., Sun, L., and Jin, R. One fits all: Power general time series analysis by pretrained lm. *arXiv preprint arXiv:2302.11939*, 2023.

A. Unified Time Series Dataset

A.1. Datasets Details

Unified Time Series Dataset (UTSD) is meticulously assembled from a blend of publicly accessible online data repositories and empirical data derived from real-world machine operations. To enhance data integrity, missing values are systematically addressed using linear interpolation techniques. We follow the unified data storage format (parquet) used in (Woo et al., 2024). For each univariate, multivariate, or irregular-sampled time series, we store them with timestamps and other meta-information in one directory using ARROW format. One dataset may composed of multiple related time series. We continue to expand the UTSD to include data from public datasets such as LOSTA[1] for zero-shot forecasting. UTSD encompasses 29 individual datasets as listed with the asterisk mark in Table 2, intricately representative of a wide range of domains.

All datasets can be classified into ten distinct domains by their source: Energy, Environment, Health, Internet of Things (IoT), Nature, Transport, Web, CloudOps, Finance, and Multiple Sources (Misc.), where the first seven domains originally come from our curated UTSD. The datasets exhibit diverse sampling frequencies, ranging from macro intervals such as yearly and quarterly to more fine-grained intervals like hourly and minutely. Notably, several datasets can demonstrate exceptionally high-frequency sampling rates, such as the MotorImagery dataset, which operates at a millisecond frequency.

In the pursuit of advanced data analysis, we have also analyzed the stationarity manifested as ADF test statistics (Elliott et al., 1996) and forecastability (Goerg, 2013). The rigorous methodologies and intricate details are elaborated in Section A.2. We utilize these statistical indicators to filter four high-quality subsets of UTSD, namely UTSD-1G, UTSD-2G, UTSD-4G, and UTSD-12G. As we expand the dataset, we continuously analyze statistical indicators and employ various methodologies to ensure the selection of high-quality datasets. LOTSA has not been sorted in this hierarchy due to its immensity.

A.2. Statistics

We analyze each dataset within our collection, examining the time series through the lenses of stationarity and forecastability. This approach allows us to characterize the level of complexity inherent to each dataset.

Stationarity The stationarity of time series is a fundamental property that can be rigorously quantified using the Augmented Dickey-Fuller (ADF) test. Notably, a larger ADF test statistic typically signifies a higher degree of non-stationarity within the time series (Elliott et al., 1996). In the context of datasets comprising multiple time series, the challenge of aligning these series arises, particularly when they vary in length. To address this, we implement a length-weighted ADF method that evaluates the stationarity of the entire dataset, taking into consideration the varying lengths of individual series. This approach ensures that the contribution of each series to the overall stationarity metric is proportional to its length, thus reflecting its relative significance within the dataset. By doing so, the length-weighted ADF provides a more granular and accurate depiction of the stationarity of the dataset, highlighting the impact of longer series on the overall stability and ensuring that shorter series do not disproportionately affect the assessment. The weighted statistic is formulated as follows:

$$T = \sum_{i=1}^{C} T_i,\ \text{ADF-Statistic}(\mathcal{D}) = \sum_{i=1}^{C} \frac{T_i}{T}\,\text{ADF-Statistic}(\mathbf{S}^{(i)}), \tag{5}$$

where $\mathbf{S}_i \in \mathbb{R}^{T_i}$ denotes the i-th series in dataset $\mathcal{D}$, T_i is the length of $\mathbf{S}_i$ and C is the number of time series of dataset $\mathcal{D}$.

Forecastability Forecastability is calculated by subtracting the entropy of the series Fourier decomposition adopted from Goerg (2013), where a higher forecastability value indicates superior predictability. Just as with the assessment of stationarity, when considering a dataset composed of multiple time series of varying lengths, it is essential to adjust the measure of forecastability to account for these differences. Therefore, we extend the concept of forecastability to a weighted version, analogous to the length-weighted ADF method, to finely tune the predictability assessment to the characteristics of each series. The weighted forecastability for a dataset can be formulated as follows:

$$T = \sum_{i=1}^{C} T_i,\ \text{Forecastability}(\mathcal{D}) = \sum_{i=1}^{C} \frac{T_i}{T}(1 - \text{Entropy}(\mathcal{F}(\mathbf{S}^{(i)}))), \tag{6}$$

[1] https://huggingface.co/datasets/Salesforce/lotsa_data

where $\mathbf{S}_i \in \mathbb{R}^{T_i}$ denotes the i-th time series in dataset $\mathcal{D}$, T_i is the length of $\mathbf{S}_i$ and C is the number of time series in dataset $\mathcal{D}$. $\mathcal{F}(\mathbf{S}^{(i)})$ denotes the Fourier decomposition of series $\mathbf{S}^{(i)}$.

Table 2: Dataset detailed descriptions. *Time Points* denotes the total number of time points aggregating from all variates if multivariate. *File Size* denotes the storage that the ARROW format of the dataset occupies. *Freq.* denotes the sampling interval of time points, where "-" indicates no timestamp or irregular interval. *ADF.* denotes the Augmented Dickey-Fuller test statistics of the dataset. *Forecast.* denotes the forecastability of the dataset. *Source* denotes the original paper or resource of the dataset.

Domain	Dataset	Time Points	File Size	Freq.	ADF.	Forecast.	Source
Energy	London Smart Meters*	166.50M	636M	Hourly	-13.158	0.173	Godahewa et al. (2021)
	Wind Farms*	7.40M	29M	4 sec	-29.174	0.811	Godahewa et al. (2021)
	Aus. Electricity Demand*	1.16M	5M	30 min	-27.554	0.730	Godahewa et al. (2021)
	BDG-2 Panther	0.92M	4M	Hourly	-6.593	0.479	Emami et al. (2023)
	BDG-2 Fox	2.32M	9M	Hourly	-9.191	0.469	Emami et al. (2023)
	BDG-2 Rat	4.73M	19M	Hourly	-6.868	0.456	Emami et al. (2023)
	BDG-2 Bear	1.48M	6M	Hourly	-11.742	0.471	Emami et al. (2023)
	Low Carbon London	9.54M	37M	Hourly	-12.366	0.134	Emami et al. (2023)
	SMART	0.10M	1M	Hourly	-10.755	0.143	Emami et al. (2023)
	IDEAL	1.26M	5M	Hourly	-11.223	0.106	Emami et al. (2023)
	Sceaux	0.03M	1M	Hourly	-14.172	0.143	Emami et al. (2023)
	Borealis	0.08M	1M	Hourly	-6.612	0.160	Emami et al. (2023)
	Buildings900K	15852.22M	60102M	Hourly	-8.412	0.357	Emami et al. (2023)
	Covid19 Energy	0.03M	1M	Hourly	-13.768	0.698	Wang et al. (2023b)
	GEF12	1.58M	6M	Hourly	-9.576	0.566	Wang et al. (2023b)
	GEF14	0.02M	1M	Hourly	-9.372	0.628	Wang et al. (2023b)
	GEF17	0.28M	1M	Hourly	-5.976	0.599	Wang et al. (2023b)
	PDB	0.04M	1M	Hourly	-6.453	0.622	Wang et al. (2023b)
	Spanish	0.07M	1M	Hourly	-13.217	0.770	Wang et al. (2023b)
	ELF	0.02M	1M	Hourly	-13.607	0.770	Wang et al. (2023b)
	KDD Cup 2022	4.73M	181M	Hourly	-17.017	0.225	Zhou et al. (2022)
	Residential Load Power	437.98M	1671M	Minutely	-37.979	0.264	Bergmeir et al. (2023)
	Residential PV Power	373.37M	1435M	Minutely	-31.389	0.421	Bergmeir et al. (2023)
Environment	AustraliaRainfall*	11.54M	45M	Hourly	-150.10	0.458	Tan et al. (2021)
	BeijingPM25Quality*	3.66M	14M	Hourly	-31.415	0.404	Tan et al. (2021)
	BenzeneConcentration*	16.34M	63M	Hourly	-65.187	0.526	Tan et al. (2021)
	China Air Quality*	34.29M	132M	Hourly	-12.602	0.529	Zheng et al. (2015)
	Beijing Air Quality*	4.62M	18M	Hourly	-15.758	0.332	Chen (2019)

Table 2 continued from previous page

Domain	Dataset	Time Points	File Size	Freq.	ADF.	Forecast.	Source
Health	MotorImagery*	72.58M	279M	0.001 sec	-3.132	0.449	Dau et al. (2019)
	SelfRegulationSCP1*	3.02M	12M	0.004 sec	-3.191	0.504	Dau et al. (2019)
	SelfRegulationSCP2*	3.06M	12M	0.004 sec	-2.715	0.481	Dau et al. (2019)
	AtrialFibrillation*	0.04M	1M	0.008 sec	-7.061	0.167	Dau et al. (2019)
	PigArtPressure*	0.62M	3M	-	-7.649	0.739	Dau et al. (2019)
	PigCVP*	0.62M	3M	-	-4.855	0.577	Dau et al. (2019)
	IEEEPPG*	15.48M	61M	0.008 sec	-7.725	0.380	Tan et al. (2021)
	BIDMC32HR*	63.59M	244M	-	-14.135	0.523	Tan et al. (2021)
	TDBrain*	72.30M	283M	0.002 sec	-3.167	0.967	Wang et al. (2023a)
	CDC Fluview ILINet	0.28M	2M	Weekly	-4.381	0.307	CDC (2017)
	CDC Fluview WHO NREVSS	0.14M	1M	Weekly	-7.928	0.233	CDC (2017)
	Project Tycho	1.35M	5M	Weekly	-8.167	0.111	van Panhuis et al. (2018)
IoT	SensorData*	165.4M	631M	0.02 sec	-15.892	0.917	Real-world machine logs
Nature	Phoneme*	2.16M	9M	-	-8.506	0.243	Dau et al. (2019)
	EigenWorms*	27.95M	107M	-	-12.201	0.393	Dau et al. (2019)
	ERA5*	96458.81M	368610M	Hourly	-7.970	0.581	Nguyen et al. (2024)
	CMIP6	104593.00M	399069M	6 H	-7.960	0.573	Nguyen et al. (2024)
	Temperature Rain*	23.25M	93M	Daily	-10.952	0.133	Godahewa et al. (2021)
	StarLightCurves*	9.46M	37M	-	-1.891	0.555	Dau et al. (2019)
	Saugeen River Flow*	0.02M	1M	Daily	-19.305	0.300	Godahewa et al. (2021)
	KDD Cup 2018*	2.94M	12M	Hourly	-10.107	0.362	Godahewa et al. (2021)
	US Births*	0.00M	1M	Daily	-3.352	0.675	Godahewa et al. (2021)
	Sunspot*	0.07M	1M	Daily	-7.866	0.287	Godahewa et al. (2021)
	Worms	0.23M	1M	0.033 sec	-3.851	0.395	Dau et al. (2019)
	Subseasonal	56.79M	217M	Daily	-12.391	0.414	Mouatadid et al. (2024)
	Subseasonal Precipitation	9.76M	38M	Daily	-13.567	0.276	Mouatadid et al. (2024)
Transport	Pedestrian Counts*	3.13M	12M	Hourly	-23.462	0.297	Godahewa et al. (2021)
	PEMS 03	9.38M	36M	5 min	-19.051	0.411	Jiang et al. (2023)
	PEMS 04	15.65M	60M	5 min	-15.192	0.494	Jiang et al. (2023)
	PEMS 07	24.92M	96M	5 min	-20.603	0.466	Jiang et al. (2023)
	PEMS 08	9.11M	35M	5 min	-14.918	0.551	Jiang et al. (2023)
	PEMS Bay	16.94M	65M	5 min	-12.770	0.704	Jiang et al. (2023)
	Los-Loop	7.09M	28M	5 min	-16.014	0.657	Jiang et al. (2023)

Table 2 continued from previous page

Domain	Dataset	Time Points	File Size	Freq.	ADF.	Forecast.	Source
Transport	Loop Seattle	33.95M	130M	5 min	-32.209	0.535	Jiang et al. (2023)
	SZ-Taxi	0.46M	2M	15 min	-5.900	0.217	Jiang et al. (2023)
	Beijing Subway	0.87M	22M	30 min	-8.571	0.219	Jiang et al. (2023)
	SHMetroy	5.07M	20M	15 min	-17.014	0.222	Jiang et al. (2023)
	HZMetro	0.38M	2M	15 min	-11.254	0.232	Jiang et al. (2023)
	Q-Traffic	264.39M	1011M	15 min	-15.761	0.490	Jiang et al. (2023)
	Taxi	55.00M	212M	30 min	-8.302	0.146	Alexandrov et al. (2020)
	Uber TLC Daily	0.05M	1M	Daily	-1.778	0.285	Alexandrov et al. (2020)
	Uber TLC Hourly	1.13M	5M	Hourly	-9.022	0.124	Alexandrov et al. (2020)
	LargeST	4452.20M	16988M	5 min	-38.020	0.436	Liu et al. (2023a)
Web	Web Traffic*	116.49M	462M	Daily	-8.272	0.299	Godahewa et al. (2021)
	Wiki-Rolling	40.62M	157M	Daily	-5.524	0.242	Alexandrov et al. (2020)
CloudOps	Alibaba Cluster Trace 2018	190.39M	2909M	5 min	-5.303	0.668	Woo et al. (2023)
	Azure VM Traces 2017	885.52M	10140M	5 min	-11.482	0.290	Woo et al. (2023)
	Borg Cluster Data 2011	1073.89M	14362M	5 min	-8.975	0.505	Woo et al. (2023)
Sales	M5	58.33M	224M	Daily	-6.985	0.247	Alexandrov et al. (2020)
	Favorita Sales	139.18M	535M	Daily	-6.441	0.097	Kaggle
	Favorita Transactions	0.08M	1M	Daily	-5.481	0.362	Kaggle
	Restaurant	0.29M	2M	Daily	-4.650	0.126	Kaggle
	Hierarchical Sales	0.21M	1M	Daily	-8.704	0.078	Mancuso et al. (2021)
Finance	GoDaddy	0.26M	2M	Monthly	-1.539	0.784	Kaggle
	Bitcoin*	0.07M	1M	Daily	-2.493	0.398	Godahewa et al. (2021)
Misc.	M1 Yearly	0.00M	1M	Yearly	-0.791	0.473	Godahewa et al. (2021)
	M1 Quarterly	0.01M	1M	Quarterly	-0.175	0.572	Godahewa et al. (2021)
	M1 Monthly	0.04M	1M	Monthly	-1.299	0.588	Godahewa et al. (2021)
	M3 Yearly	0.02M	1M	Yearly	-0.850	0.524	Godahewa et al. (2021)
	M3 Quarterly	0.04M	1M	Quarterly	-0.897	0.624	Godahewa et al. (2021)
	M3 Monthly	0.1M	1M	Monthly	-1.954	0.635	Godahewa et al. (2021)
	M3 Other	0.01M	1M	-	-0.568	0.801	Godahewa et al. (2021)
	M4 Yearly	0.84M	4M	Yearly	-0.036	0.533	Godahewa et al. (2021)
	M4 Quarterly	2.214M	10M	Quarterly	-0.745	0.696	Godahewa et al. (2021)
	M4 Monthly	10.38M	41M	Monthly	-1.358	0.665	Godahewa et al. (2021)

TABLE 2 CONTINUED FROM PREVIOUS PAGE

DOMAIN	DATASET	TIME POINTS	FILE SIZE	FREQ.	ADF.	FORECAST.	SOURCE
MISC.	M4 WEEKLY	0.37M	2M	WEEKLY	-0.533	0.644	GODAHEWA ET AL. (2021)
	M4 DAILY	9.96M	39M	DAILY	-1.332	0.841	GODAHEWA ET AL. (2021)
	M4 HOURLY	0.35M	2M	HOURLY	-2.073	0.532	GODAHEWA ET AL. (2021)

[*] THE ASTERISK MARKS THE DATASET THAT ORIGINALLY BELONGS TO UTSD.

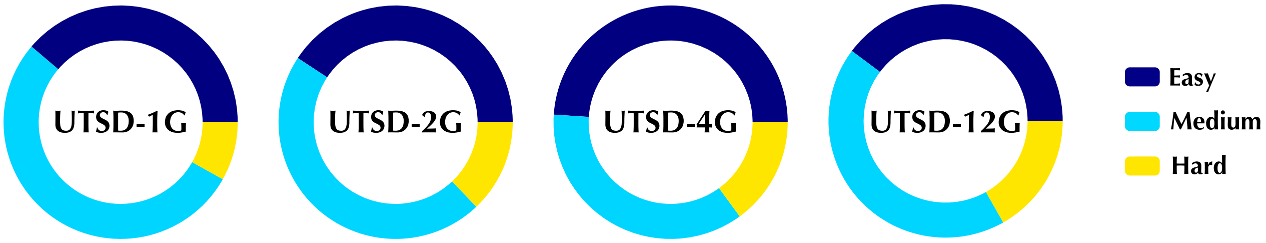

Figure 16. Dataset complexity in each hierarchy of UTSD.

A.3. UTSD Composition Analysis

UTSD is constructed with hierarchical capacities, namely UTSD-1G, UTSD-2G, UTSD-4G, and UTSD-12G, where each smaller dataset is a subset of the larger ones. We adhere to the principle of progressively increasing the complexity and pattern diversity. Hierarchical structuring allows for a nuanced analysis that accounts for different levels of data granularity and complexity, ensuring that the pattern diversity is maintained across each hierarchy of the dataset. This approach not only facilitates a comprehensive evaluation across different scales but also ensures that each subset within the larger dataset offers a unique and incrementally challenging perspective, thus contributing to a more scalable pre-training.

Dataset Complexity We conduct a comprehensive analysis of individual datasets to obtain the stationarity and forecastability measures and construct the UTSD hierarchically regarding these indicators. Code for calculating the statistics is provided in the repository of UTSD. Consequently, based on the ADF-Statistic of each dataset, we categorized the predictive difficulty of the datasets into three levels: Easy, Medium, and Hard. The criteria are listed as follows:

- **Easy**: $\text{ADF-Statistic} < -15.00$;
- **Medium**: $-15.00 \leq \text{ADF-Statistic} < -5.00$;
- **Hard**: $-5.00 \leq \text{ADF-Statistic}$.

For excessively long datasets in the temporal dimension, we additionally adopt the forecastability to assess the complexity of time series across different periods. As the capacity of UTSD increases, the periods with low forecastability will be further incorporated correspondingly. In a nutshell, larger datasets contain a greater proportion of challenging tasks as shown in Figure 16, thereby escalating the complexity of the pre-training process. The hierarchy reflects an increase in the difficulty of patterns as the dataset size grows. This approach enables a structured examination of the learning challenges presented by different dataset sizes, underlining the intricate balance between data volume and pre-training difficulty.

Pattern Diversity Each dataset within the UTSD collection demonstrates unique patterns, highlighting the importance of maintaining pattern diversity. We build the UTSD dataset in a top-down manner, ensuring that each hierarchy within UTSD comprehensively represents all individual datasets and contains as many patterns as possible. As shown in Figure 17, we select several representative datasets for visualization analysis:

- **AtrialFibrillation**: The dataset showcases a fluctuating trend with minimal seasonality. This pattern is an indicator of irregular heart rhythm characteristics, typical in medical recordings related to cardiac health. Such fluctuations, lacking a clear seasonal pattern, are crucial for understanding the unpredictable nature of atrial fibrillation.

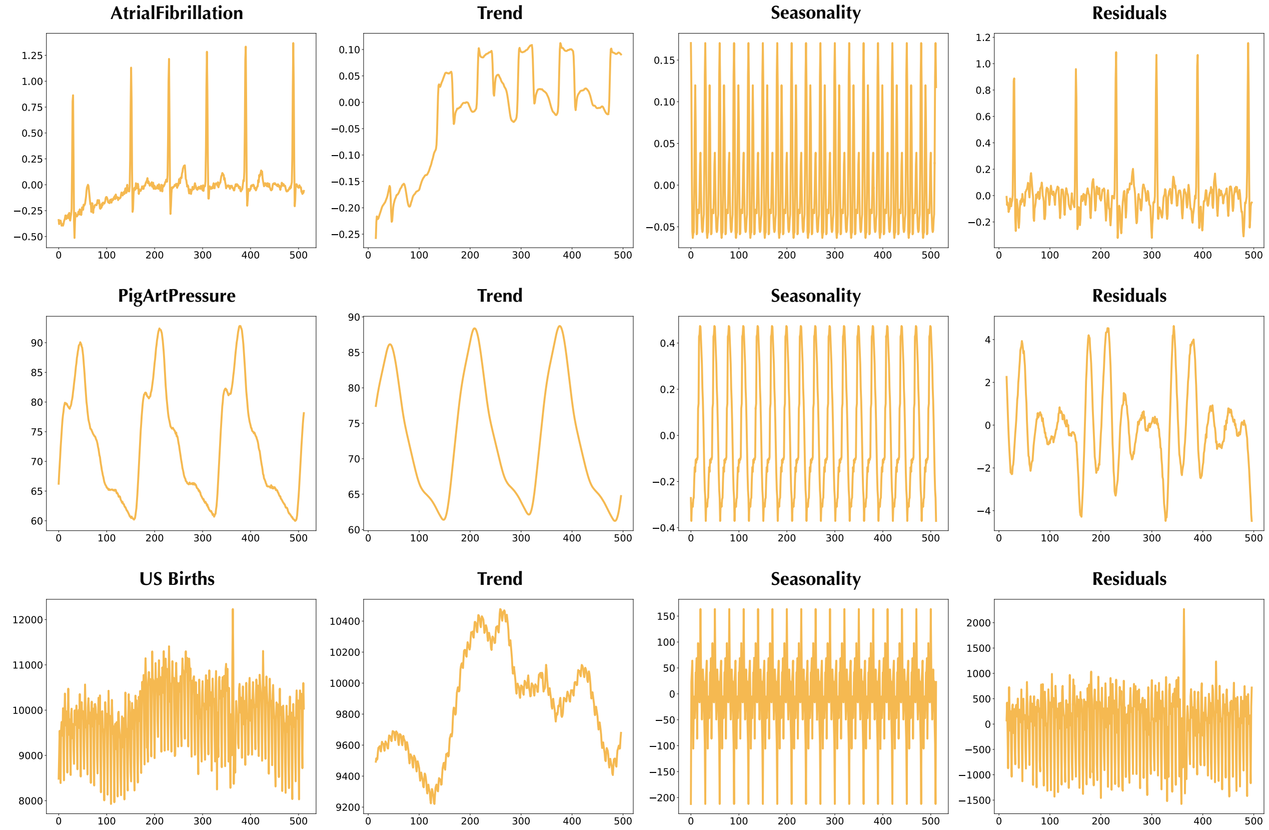

Figure 17. Visualization of representative patterns in UTSD. Each time series is decomposed into trend, seasonal, and residual components.

- **PigArtPressure**: The dataset reveals a fluctuating trend interspersed with notable seasonality. This pattern is representative of the physiological variations in blood pressure that can occur due to environmental or internal factors. The presence of both fluctuating trends and seasonality in this dataset underscores the complex nature of biological data.

- **US Births**: The dataset distinctly exhibits a clear trend alongside pronounced seasonality. This pattern is characteristic of demographic data, where trends and seasonal variations can reflect socio-cultural factors and environmental influences. The consistent trend and seasonality in birth rates provide insights into population dynamics and reproductive behaviors.

To avoid selecting trivial temporal variations and provide a comprehensive representation of the varied patterns inherent in the individual datasets, we employ a downsampling technique for individual datasets. For those with a larger number of variates, we selectively choose representative variates that best encapsulate the distinct patterns of respective datasets. Similarly, for datasets with considerable temporal length, we resample them by the representative period. This methodical selection process ensures that the integrity and distinctive characteristics of each dataset are preserved, thereby maintaining the diversity of patterns across the hierarchical structure of the UTSD dataset.

A.4. Experiments

Forecasting benchmarks In the field of time series forecasting, several classical datasets such as ETT (Zhou et al., 2021), ECL (Wu et al., 2021), Traffic (Wu et al., 2021) and Weather (Wu et al., 2021) have become widely recognized benchmarks for evaluating model performance. However, the variability in several datasets, such as ECL, is relatively homogeneous, and they do not adequately address aspects such as non-stationarity and predictability when assessing the strengths and weaknesses of models. Consequently, the development of a new benchmark is essential. Therefore, we have carefully considered factors such as domain, number of variables, frequency, non-stationarity, and predictability, and have selected a subset from the UTSD as the new benchmark. The datasets we have selected are presented in Table 3. Furthermore, we have evaluated our model along with other baseline models on these benchmarks. The results are presented in Table 4. Admittedly, relying solely on these benchmarks is not sufficient to comprehensively assess model performance. We also look forward to the proposal of more diverse and comprehensive benchmarks in the future.

Table 3. Benchmark detailed descriptions. *Time Point* denotes the total number of time points aggregating from all variates if multivariate. *Frequency* denotes the sampling interval of time points, where "-" indicates no timestamp or irregular interval. *ADF Statistic* denotes the Augmented Dickey-Fuller test statistics of the dataset. *Forecastability* denotes the forecastability of the dataset.

Domain	Dataset	Time Points	Variates	Frequency	ADF Statistic	Forecastability
Environment	AustraliaRainfall	11.54M	3	Hourly	-150.10	0.458
Transport	PedestrianCounts	0.08M	1	Hourly	-23.462	0.297
IoT	SensorData	3.24M	18	0.002 Sec	-15.892	0.917
Health	BIDMC32HR	0.04M	1000	-	-14.135	0.523

Table 4. Forecasting results on well-acknowledged deep forecasters and Timer, where Timer is pre-trained on the held-out datasets and then all models are superwisedly trained on the four datasets in the 672-pred-96 setting.

Models	**Timer**		PatchTST		iTransformer		DLinear	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
AustraliaRainfall	**0.800**	**0.720**	0.802	**0.720**	**0.800**	0.800	0.804	0.804
PedestrianCounts	**0.054**	**0.133**	0.058	0.153	0.056	0.143	0.060	0.149
SensorData	**0.049**	0.094	0.056	0.094	0.052	**0.091**	0.057	0.111
BIDMC32HR	**0.030**	**0.062**	0.188	0.284	0.159	0.249	0.320	0.409

Domain transfer To investigate the domain partitioning of UTSD, we use different domains of UTSD as the source and adapt the trained model to different target datasets to establish in-domain and out-of-domain transfer. The results in Table 5 indicate that in-domain transfer can further enhance the downstream performance. Additionally, as the number of downstream data samples increases, the relative improvement of pre-training will gradually diminish, and even lead to negative transfer in some out-of-domain scenarios. It provides a promising direction to develop domain-specific models.

Table 5. In-domain and out-of-domain forecasting results by pre-training on the source domain and fine-tuning on the target dataset under different data scarcity. ECL and Weather belong to the Energy and Nature domains respectively.

Target Dataset	Weather						ECL					
Source Domain	From Scratch		Energy		**Nature**		From Scratch		Nature		**Energy**	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
5% Target	0.229	0.279	0.171	0.220	**0.162**	**0.212**	0.179	0.277	0.165	0.269	**0.141**	**0.238**
20% Target	0.185	0.238	0.160	0.212	**0.153**	**0.202**	0.145	0.243	0.140	0.238	**0.133**	**0.228**
100% Target	0.158	0.209	0.152	0.199	**0.151**	**0.198**	**0.130**	0.224	0.132	0.224	0.131	**0.223**

B. Implementation Details

B.1. Pre-training

Based on the constructed UTSD datasets of different sizes and difficulties in the unified single series sequence (S3) format, Timer is pre-trained with increasing data sizes and model parameters to validate the scalability. Detailed configurations and parameter counts of the pre-trained models involved in this paper are provided in Table 6.

Table 6. Detailed model configurations of Timer and corresponding parameter counts. The number of heads for models is fixed as 8.

SCENARIO	MODEL DIM. SCALE-UP				LAYER NUMBER. SCALE-UP				OTHERS	
SCALE	3M	13M	29M	51M	1M	2M	3M	4M	38M	67M
LAYERS	6	6	6	6	2	4	6	8	8	8
MODEL DIM.	256	512	768	1024	256	256	256	256	768	1024
FFN DIM.	512	1024	1536	2048	512	512	512	512	1536	2048
PARAMETERS	3.21M	12.72M	28.51M	50.59M	1.10M	2.16M	3.21M	4.27M	37.97M	67.40M

All experiments are implemented in PyTorch (Paszke et al., 2019) and trained using NVIDIA A100 Tensor Core GPU. We use AdamW (Kingma & Ba, 2015) as the optimizer and cosine annealing algorithm for learning rate decay. The base learning rate is 5×10^{-5}, and the final learning rate is 2×10^{-6} . The decay steps are proportional to the number of training steps of 10 epochs. During pre-training, we use $N = 15$ as the number of tokens, and the batch size is set to 8192.

During the pre-training on the UTSD-1G to UTSD-4G, we adopt a global shuffle strategy by loading the whole time series into the memory. Due to the much greater data scale of UTSD-12G compared to any commonly used time series dataset in the past, it is difficult to load all 12GB of the pre-training dataset into memory for global shuffling. Therefore, we use a local shuffle strategy, which randomly selects and divides the 12GB pre-training dataset into three 4G subsets in the storage space through file selection and segmentation, and then takes turns loading them into memory for pre-training with global steps. In this strategy, we also ensure the continuity of learning rate decay.

B.2. Downstream Tasks

We introduce the details of downstream experiments and present the generative scheme for each task, including time series forecast, imputation, and anomaly detection. Configurations for downstream adaptation are listed in Table 8. Corresponding detailed results are provided in Section C. And showcases of downstream tasks are shown in Figure 19, 20, and 21.

Forecasting The downstream forecasting task is tested on the real-world datasets, including (1) ETT (Zhou et al., 2021) contains 7 variates of power transformers, with the period from July 2016 to July 2018, including four subsets and sampling intervals of one hour and fifteen minutes. (2) ECL (Wu et al., 2021) mainly consists of hourly electricity consumption data from 321 customers (3) Traffic (Wu et al., 2021) collected hourly road occupancy rates measured by 862 sensors on the San Francisco Bay Area highway from January 2015 to December 2016. (4) Weather (Wu et al., 2021) consists of 21 meteorological variates collected every 10 minutes from the Max Planck Institute of Biogeochemistry meteorological station in 2020. (5) PEMS contains California public transportation network data collected through a 5-minute window with the same four common subsets (PEMS03, PEMS04, PEMS07, PEMS08) used in SCINet (Liu et al., 2022).

We adopt the autoregressive generation training objective (Bengio et al., 2000) for downstream forecasting datasets in the fine-tuning stage. Specifically, we divide the lookback length into $N = 7$ tokens with the segment length $S = 96$. The model naturally outputs N next tokens, which we calculate the mean squared error (MSE) of the N tokens with corresponding ground truth and backpropagate the loss. During inference, we conduct iterative multi-step forecasting by concatenating the forecasted result with the lookback series and repeatedly adopting the model to generate the next token until the total length of predicted tokens reaches the expected length. If exceeding the predicted length, we will crop the excess value of the end.

For constructing data-scarce scenarios, we perform retrieval with the uniform interval in the training split according to the sampling ratio and conduct random shuffling at the end of each epoch to train the model. The construction pipeline with the fixed random seed ensures the reproducibility of our experimental results. In order to maintain comparability with previous benchmarks, we keep the same validation and testing sets of original downstream datasets and train the baseline model and Timer with the same set of training samples.

Imputation Considering the real-world scenario that missing values at time points often appear in succession, we adjust the previous point-level imputation proposed by TimesNet (Wu et al., 2022) and increase the difficulty of the task, that is, changing the masked unit from point to time series segment. The protocol poses challenges to recovering successive points,

Table 7. Downstream forecasting dataset descriptions. *Split* denotes the number of time points in (train, validation, test) splits. *Frequency* denotes the sampling interval of time points. *Information* denotes the domain in which the dataset belongs to.

Dataset	Variate	Split	Frequency	Information
ETTh1, ETTh2	7	(8545, 2881, 2881)	Hourly	Electricity
ETTm1, ETTm2	7	(34465, 11521, 11521)	15min	Electricity
ECL	321	(18317, 2633, 5261)	Hourly	Electricity
Traffic	862	(12185, 1757, 3509)	Hourly	Transportation
Weather	21	(36792, 5271, 10540)	10min	Weather
PEMS03	358	(15617, 5135, 5135)	5min	Transportation
PEMS04	307	(10172,3375, 3375)	5min	Transportation
PEMS07	883	(16911, 5622, 5622)	5min	Transportation
PEMS08	170	(10690, 3548, 3548)	5min	Transportation

which underscore higher demands for the model capacity to restore a span of series variations. Concretely, for a time series consisting of $N = 8$ segments with a length of 24, we randomly mask several segments as zeros except for the first segment, ensuring that the first segment is observed by the model to learn about initial series variations for imputation. For the training objective of downstream adaption, we adopt the denoising autoencoding (Raffel et al., 2020), which takes the masked parts as special tokens and unmasked segments as tokens as the model input. Due to the generative capability of Timer acquired by pre-training, we regard outputted tokens as the next predicted tokens and backpropagate the reconstruction error between the generated next token of the masked segment with the ground truth. During inference, we take the MSE of the masked segments as the indicator to evaluate the imputation performance. Based on the above protocol, we conduct the imputation task on the same datasets of the forecasting task in Table 7.

Table 8. Detailed explanation of model hyperparameters and corresponding parameter quantities. We adopt the learning rate schedule strategy with exponential decay at a base of 0.5 under all three downstream tasks.

Tasks	Model Hyper-parameter				Training Process			
	L_{min}	L_{max}	d_{min} †	d_{max} †	LR*	Loss	Batch Size	Epochs
Forecasting	2	8	256	2048	3e-5	MSE	2048	10
Imputation	4	4	256	256	3e-5	MSE	32	10
Anomaly Detection	4	4	256	256	3e-5	MSE	128	10

∗ LR means the initial learning rate.

Anomaly detection For anomaly detection, prevalent protocols represented by Anomaly Transformer (Xu et al., 2021) and TimesNet (Wu et al., 2022) adopt the reconstructive approach that learns a feature extractor to reconstruct raw series, and the output is regarded as standard values. With all the mean squared errors between the standard and input series from the datasets, a specific threshold with the given quantile is determined to label the anomalies.

Considering the prevalent scenarios of anomaly detection by monitoring real-time measurements, the quick judgment of on-the-fly time series anomaly can be more practical in the real world. Therefore, we propose a predictive protocol of anomaly detection based on generative models. Concretely, we use the observed segments to predict the future segment, and the predicted segment will be established as the standard to be compared with the actual value received. We adopt the UCR Anomaly Archive proposed by Wu & Keogh (2021). The task is to find the position of an anomaly in the test series based on a single normal series for training, which is an extremely data-scarce scenario with only one available sample. For downstream adaption, we adopt the same next token prediction as the pre-training, that is, training Timer with the lookback series containing $N = 7$ segments of the length $S = 96$ to generate the next token with length 96, which is regarded as the standard value. After training, we record the MSE of all segments in the test set and sort them in descending order. We find the first segment hit the anomaly interval labeled in the dataset within the first α quantile, and we record the quantile. Based on the above protocol, the real-time judgment ability of the model for sudden anomalies can be predictively examined. Detailed quantiles of Timer in 250 tasks are provided in Table 15. With more complex time series anomalies introduced in UCR Anomaly Archive, we hope to establish a reasonable and challenging benchmark in the field of anomaly detection.

Zero-shot forecasting We conduct zero-shot forecasting experiments on seven datasets from iTransformer (Liu et al., 2023b). Notably, PEMS datasets are not included, as they have already appeared in the LOSTA dataset for pre-training. We apply the same data-split strategy as Autoformer (Wu et al., 2021) and calculate the averaged MSE of all predict-96 windows in the test split. We evaluate five open-source large time series models, including Timer, Moiria (Woo et al., 2024), TimesFM (Das et al., 2023b), Chronos (Ansari et al., 2024), and MOMENT (Goswami et al., 2024). We further assess the qualities in Table 9, which includes more LTSMs and summarizes several attributes and abilities of large models.

- **MOMENT**: MOMENT[2] trained by masking modeling is applied to zero-shot forecasting by concatenating the lookback series with a mask with the length to be predicted. The mask through the model is regarded as the prediction.
- **Chronos**: Chronos[3] is a probabilistic forecaster. *Chronos-S1* means sampling one prediction trajectory and *Chronos-S20* means sampling 20 trajectories and using the average trajectory.
- **TimesFM**: We use the official checkpoint from HuggingFace[4], which supports various input and output lengths.
- **Moiria**: The Moiria family[5] has three different sizes, labeled as *Moiria-S*, *Moiria-M*, and *Moiria-L*.
- **Timer**: We provide three versions with increased scopes of pre-training. *Timer-1B* is pre-trained on UTSD; *Timer-16B* is pre-trained on UTSD and Buildings900K (Emami et al., 2023); and *Timer-28B* is pre-trained on UTSD and LOTSA.

Table 9. Quality evaluation of large time series models. *Architecture* denotes the Transformer category. *Model size* presents the parameter counts. *Token type* presents the graininess of time series tokens. *Context length* means the maximum/fixed input length of the model.

Method	**Timer (Ours)**	Moirai (2024)	MOMENT (2024)	Chronos (2024)	Lag-LLama (2023)	TimesFM (2023b)	TimeGPT-1 (2023)
Architecture	Decoder	Encoder	Encoder Decoder	Encoder Decoder	Decoder	Decoder	Encoder Decoder
Model size	29M, 50M, 67M	14M, 91M, 311M	40M, 125M 385M	20M, 46M, 200M, 710M	200M	17M, 70M, 200M	UnKnown
Supported tasks	Forecast Imputation Detection	Forecast	Forecast Imputation Classification Detection	Forecast	Forecast	Forecast	Forecast Detection
Pre-training Scale	28B	27.65B	1.13B	84B	0.36B	100B	100B
Token type	Segment	Segment	Segment	Point	Point	Segment	Segment
Context length	$\leq$1440	$\leq$5000	= 512	$\leq$512	$\leq$1024	$\leq$512	Unknown
Variable length	True	True	False	True	True	True	True
Probabilistic	False	True	False	True	True	True	True

[2] https://huggingface.co/AutonLab/MOMENT-1-large

[3] https://huggingface.co/amazon/chronos-t5-large

[4] https://huggingface.co/google/timesfm-1.0-200m

[5] https://huggingface.co/collections/Salesforce/moirai-10-r-models-65c8d3a94c51428c300e0742

C. Full Results

C.1. Time Series Forecasting

We provide all the results of the forecasting task in Figure 6. As shown in Table 10, we include six representative real-world datasets, demonstrating that Timer achieves state-of-the-art forecasting performance and the large-scale pre-training helps to alleviate performance degradation as the available downstream samples decrease.

Table 10. Full forecasting results of Timer obtained by training from scratch (None) and fine-tuning from UTSD-12G pre-trained model. The bold values we use indicate that the pre-trained model results have positive benefits compared to from scratch. We attach the current state-of-the-art results as *SOTA* in this table, including PatchTST (Nie et al., 2022) on ETTh1 and Weather, as well as iTransformer (Liu et al., 2023b) on ECL, Traffic, PEMS03, and PEMS04. We adopt the unified lookback length as 672 and the forecast length as 96.

Dataset	ETTh1		ECL		Traffic		Weather		PEMS03		PEMS04	
Pre-trained	None	12G	None	12G	None	12G	None	12G	None	12G	None	12G
100%	0.363	**0.358**	0.132	0.136	0.352	**0.351**	0.165	**0.154**	0.126	**0.118**	0.125	**0.107**
75%	0.364	**0.358**	0.132	0.137	0.353	**0.351**	0.162	**0.157**	0.124	**0.114**	0.126	**0.110**
50%	0.370	**0.356**	0.132	0.135	0.356	**0.352**	0.161	**0.151**	0.129	**0.114**	0.131	**0.110**
25%	0.387	**0.359**	0.135	**0.134**	0.368	**0.352**	0.162	**0.153**	0.133	**0.114**	0.141	**0.117**
20%	0.385	**0.359**	0.137	**0.134**	0.372	**0.352**	0.166	**0.151**	0.135	**0.116**	0.145	**0.120**
15%	0.391	**0.360**	0.141	**0.134**	0.379	**0.353**	0.174	**0.152**	0.138	**0.118**	0.152	**0.123**
10%	0.426	**0.361**	0.144	**0.133**	0.387	**0.353**	0.182	**0.152**	0.140	**0.120**	0.165	**0.126**
5%	0.426	**0.362**	0.154	**0.132**	0.407	**0.361**	0.198	**0.151**	0.158	**0.125**	0.195	**0.135**
4%	0.424	**0.362**	0.161	**0.135**	0.416	**0.366**	0.208	**0.152**	0.166	**0.127**	0.210	**0.138**
3%	0.427	**0.363**	0.169	**0.134**	0.431	**0.369**	0.218	**0.153**	0.180	**0.131**	0.234	**0.143**
2%	0.427	**0.363**	0.186	**0.137**	0.467	**0.380**	0.230	**0.159**	0.201	**0.137**	0.257	**0.152**
1%	0.428	**0.366**	0.215	**0.140**	0.545	**0.390**	0.246	**0.166**	0.249	**0.151**	0.320	**0.172**
SOTA	0.370		0.129		0.360		0.149		0.132		0.115	

C.2. Imputation

In this section, we provide the detailed results of the imputation task, including Timer trained from scratch and adapting pre-trained models with 5% available samples in Table 11, 20% samples in Table 12, and full samples in Table 13 on the downstream task. We also report the results of TimesNet at the above three ratios in Table 14. Based on the result, we provided an improvement in imputation performance before and after pre-training with $\{5\%, 20\%, 100\%\}$ samples in Figure 8 and 18, reflecting the benefits of autoregressive pre-training in segment-wise imputation task.

C.3. Anomaly Detection

In this section, we provide detailed results of anomaly detection in Table 15, including the results of Timer from scratch and pre-trained. We conducted experiments on all 250 datasets of UCR Anomaly Archive and calculated the corresponding α quantiles. The results show that the pre-trained Timer can detect time series anomalies with smaller α on most datasets.

C.4. Scalability

We provide detailed downstream forecasting results conducted on PEMS subsets with the scaling of model size (Figure 10) and data size (Figure 11). As shown in Table 16, it supports the scalability of our decoder-only Timer trained in GPT-style,

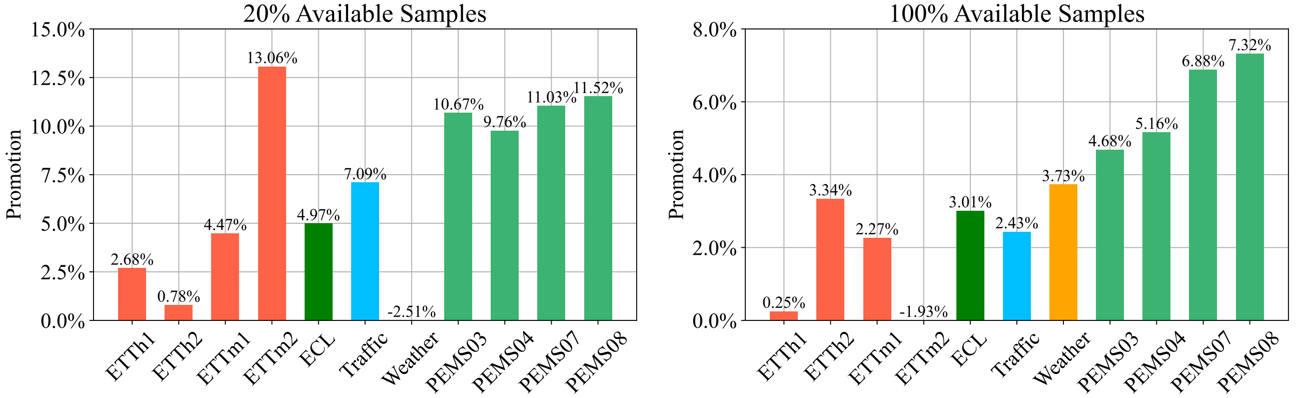

Figure 18. Pre-training benefit of Timer on the downstream imputation task with 20% and 100% available samples. Complete results details used to calculate the pre-training benefits relative to training from scratch in the figures are listed in Table 11-14.

following the scaling law (Kaplan et al., 2020) towards large time series models.

C.5. Zero-shot forecasting

In this section, we provide detailed results of zero-shot forecasting in Table 18. We conducted experiments on seven datasets that are not included in the pre-training corpora of LTSMs. The results show that the top-ranked LTSMs are Timer, Moiria, and TimesFM. The performance of probabilistic forecaster Chronos can be improved by sampling more trajectories. It can still be an issue that scaling behavior is not evident on some datasets in the zero-shot scenario, and the failure of multi-step prediction can also appear in some models, indicating the development of zero-shot LTSMs is still in the early stage.

D. Showcase

To present a clear performance of our proposed Timer, we provide visualizations for downstream forecasting, imputation, and anomaly detection tasks in Figure 19, 20 and 21. The forecasting and imputation contain experimental results at different sample ratios. For anomaly detection, we provide the position of the anomaly and the generated normal series by Timer.

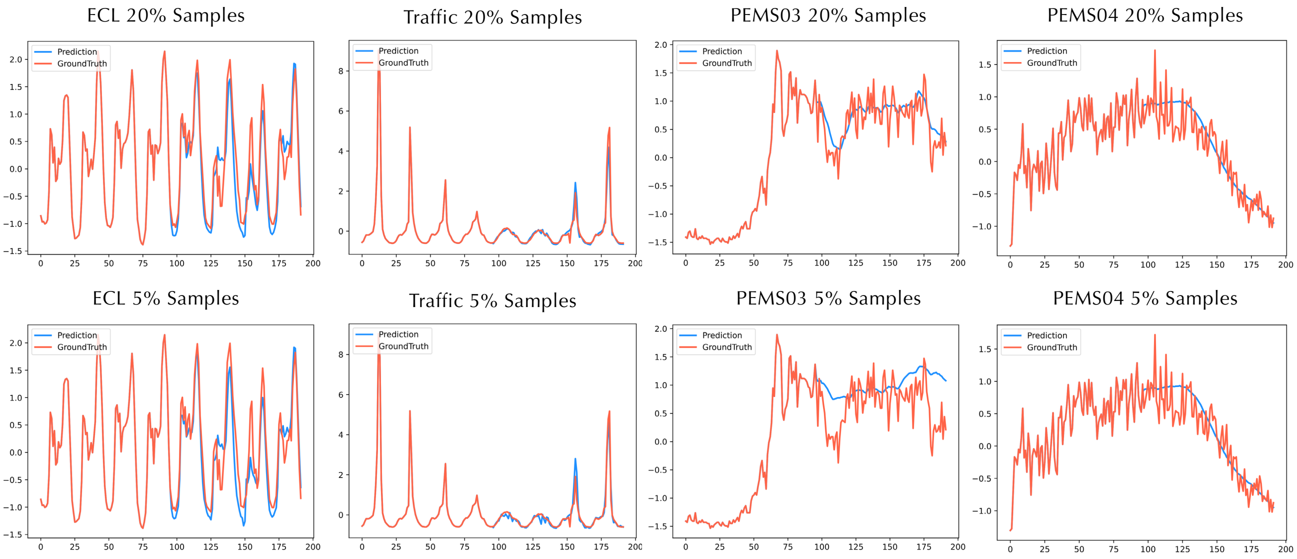

Figure 19. Visualization of input-672-predict-96 forecasting results of Timer trained with 5% and 20% samples.

Table 11. Downstream imputation with 5% samples. Pre-training benefit $\Delta\%$ is calculated as the ratio of decreased imputing error in MSE. In the case of 5% samples, our pre-trained model outperforms TimesNet (Table 14) in all 44 settings on datasets and masked ratios.

Mask Ratio	12.5%			25.0%			37.5%			50.0%		
Pre-trained	None	12G	$\Delta\%$	None	12G	$\Delta\%$	None	12G	$\Delta\%$	None	12G	$\Delta\%$
ETTh1	0.301	0.292	**+3.08**	0.313	0.299	**+4.46**	0.322	0.307	**+4.59**	0.325	0.325	0.00
ETTh2	0.172	0.168	**+2.64**	0.182	0.180	**+1.26**	0.197	0.190	**+3.22**	0.216	0.215	**+0.47**
ETTm1	0.397	0.347	**+12.52**	0.403	0.332	**+17.72**	0.428	0.374	**+12.77**	0.473	0.425	**+10.13**
ETTm2	0.118	0.116	**+1.59**	0.127	0.121	**+4.69**	0.134	0.131	**+2.22**	0.147	0.144	**+1.99**
ECL	0.152	0.140	**+7.67**	0.162	0.150	**+7.27**	0.172	0.161	**+6.76**	0.185	0.174	**+6.17**
Traffic	0.538	0.460	**+14.60**	0.567	0.487	**+14.14**	0.598	0.520	**+13.16**	0.633	0.558	**+11.91**
Weather	0.113	0.117	-3.18	0.116	0.114	**+2.31**	0.128	0.124	**+3.28**	0.155	0.136	**+12.42**
PEMS03	0.160	0.135	**+15.78**	0.196	0.168	**+14.60**	0.257	0.223	**+13.51**	0.354	0.306	**+13.49**
PEMS04	0.193	0.161	**+16.80**	0.238	0.202	**+15.30**	0.305	0.258	**+15.28**	0.410	0.348	**+15.14**
PEMS07	0.166	0.139	**+16.19**	0.210	0.183	**+12.89**	0.278	0.243	**+12.72**	0.378	0.326	**+13.76**
PEMS08	0.185	0.157	**+15.33**	0.232	0.195	**+15.98**	0.303	0.265	**+12.75**	0.417	0.362	**+13.26**

E. Limitations

UTSD is constructed with hierarchical capacities. Though it is helpful to study the scalability of the model, it is not big enough since we have witnessed recent work claims the pre-training on ten and even a hundred billion time points. Therefore, we advocate for the ongoing expansion of data infrastructure while upholding high quality and hierarchy, which may significantly advance the time series community. In terms of the method, this work aims at an early but important development of large models. Despite the generalization, scalability, and task-generality that Timer has achieved, time series classification has not been unified in our generative formulations and Timer does not yet support probabilistic forecasting and specially adapts for multiple variables. It also leaves for better zero-shot generalization and advanced abilities, such as in-context learning and multi-modality, which are scheduled to be developed by ever-large pre-training.

F. Societal Impacts

Real-world applications This paper develops large models for the field of time series. We present a general-purpose time series analysis model to handle data-scarce scenarios. Given the state-of-the-art performance of Timer, this model may be applied to many real-world applications, which helps our society prevent risks in advance and make better decisions with limited available samples. Our paper mainly focuses on scientific research and has no obvious negative social impact.

Academic research In this paper, we release a high-quality dataset for scalable pre-training. Different from prior works, the dataset is not merely aggregation but follows deftly curation. Based on it, the research on scalable time series architectures and pre-training techniques can be facilitated. Towards large time series models, the proposed Timer shows its generalization and versatility in many tasks. The regime of generative pre-training and autoregression can be instructive for future research.

Table 12. Downstream imputation with 20% samples. Pre-training benefit Δ% is calculated as the ratio of decreased imputing error in MSE. In the case of 20% samples, our pre-trained model outperforms TimesNet in 86.4% of 44 settings on datasets and masked ratios.

Mask Ratio	12.5%			25.0%			37.5%			50.0%		
Pre-trained	None	12G	Δ%	None	12G	Δ%	None	12G	Δ%	None	12G	Δ%
ETTh1	0.289	0.278	**3.83**	0.293	0.287	**1.91**	0.305	0.297	**2.56**	0.322	0.314	**2.44**
ETTh2	0.168	0.166	**1.21**	0.180	0.178	**1.02**	0.192	0.190	**0.73**	0.208	0.208	**0.17**
ETTm1	0.349	0.328	**5.98**	0.335	0.326	**2.72**	0.378	0.360	**4.84**	0.426	0.407	**4.34**
ETTm2	0.139	0.133	**4.77**	0.158	0.123	**22.30**	0.176	0.136	**22.95**	0.146	0.143	**2.22**
ECL	0.136	0.130	**5.01**	0.146	0.138	**5.30**	0.157	0.149	**5.04**	0.170	0.162	**4.54**
Traffic	0.451	0.420	**6.89**	0.481	0.446	**7.26**	0.513	0.477	**7.09**	0.550	0.511	**7.10**
Weather	0.125	0.129	-3.40	0.125	0.147	-17.46	0.154	0.125	**18.77**	0.141	0.153	-7.93
PEMS03	0.134	0.120	**10.41**	0.169	0.150	**11.35**	0.221	0.198	**10.61**	0.305	0.273	**10.32**
PEMS04	0.162	0.146	**9.93**	0.203	0.184	**9.61**	0.262	0.236	**9.86**	0.354	0.320	**9.65**
PEMS07	0.140	0.125	**10.96**	0.182	0.162	**10.85**	0.240	0.214	**10.77**	0.327	0.290	**11.57**
PEMS08	0.155	0.139	**10.39**	0.198	0.174	**12.11**	0.268	0.236	**11.94**	0.366	0.324	**11.63**

Table 13. Downstream imputation with 100% samples. Pre-training benefit Δ% is calculated as the ratio of decreased imputing error in MSE. In the case of 100% samples, our pre-trained model outperforms TimesNet in 56.8% of 44 settings on datasets and masked ratios.

Mask Ratio	12.5%			25.0%			37.5%			50.0%		
Pre-trained	None	12G	Δ%	None	12G	Δ%	None	12G	Δ%	None	12G	Δ%
ETTh1	0.274	0.273	**0.34**	0.283	0.283	-0.04	0.295	0.294	**0.52**	0.313	0.312	**0.17**
ETTh2	0.207	0.177	**14.44**	0.186	0.186	-0.49	0.192	0.195	-1.18	0.210	0.209	**0.57**
ETTm1	0.342	0.352	-3.04	0.359	0.345	**3.87**	0.400	0.371	**7.09**	0.418	0.413	**1.15**
ETTm2	0.149	0.161	-8.01	0.153	0.171	-11.46	0.173	0.176	-1.53	0.183	0.158	**13.27**
ECL	0.125	0.122	**2.98**	0.134	0.130	**3.06**	0.144	0.139	**3.12**	0.157	0.152	**2.87**
Traffic	0.402	0.392	**2.50**	0.424	0.414	**2.48**	0.454	0.443	**2.46**	0.488	0.477	**2.29**
Weather	0.144	0.157	-8.67	0.159	0.146	**8.01**	0.162	0.147	**9.41**	0.168	0.158	**6.15**
PEMS03	0.113	0.108	**4.65**	0.143	0.135	**5.30**	0.188	0.179	**4.88**	0.258	0.248	**3.90**
PEMS04	0.142	0.134	**5.23**	0.176	0.166	**5.39**	0.227	0.216	**5.24**	0.311	0.296	**4.77**
PEMS07	0.121	0.114	**5.81**	0.155	0.144	**6.50**	0.204	0.189	**7.36**	0.277	0.256	**7.85**
PEMS08	0.137	0.129	**5.89**	0.169	0.157	**7.29**	0.224	0.206	**7.70**	0.314	0.288	**8.39**

Table 14. Full results of downstream imputation of TimesNet under different data scarcities as the baseline.

Sample Ratio	5%				20%				100%			
Mask Ratio	12.5%	25.0%	37.5%	50.0%	12.5%	25.0%	37.5%	50.0%	12.5%	25.0%	37.5%	50.0%
ETTh1	0.676	0.671	0.678	0.682	0.684	0.687	0.675	0.679	0.284	0.296	0.269	0.289
ETTh2	0.258	0.249	0.272	0.276	0.252	0.245	0.251	0.268	0.178	0.199	0.219	0.253
ETTm1	0.665	0.734	0.441	0.483	0.254	0.344	0.314	0.444	0.185	0.232	0.273	0.373
ETTm2	0.138	0.135	0.143	0.153	0.104	0.107	0.119	0.148	0.084	0.090	0.096	0.112
ECL	0.226	0.222	0.230	0.230	0.221	0.224	0.226	0.231	0.200	0.207	0.209	0.211
Traffic	0.802	0.794	0.801	0.809	0.801	0.791	0.798	0.805	0.773	0.775	0.624	0.565
Weather	0.155	0.141	0.162	0.168	0.135	0.124	0.132	0.157	0.104	0.114	0.111	0.127
PEMS03	0.173	0.192	0.239	0.321	0.156	0.291	0.254	0.318	0.142	0.148	0.195	0.273
PEMS04	0.215	0.243	0.291	0.379	0.179	0.222	0.266	0.350	0.123	0.167	0.210	0.285
PEMS07	0.166	0.195	0.247	0.335	0.161	0.195	0.253	0.310	0.113	0.142	0.191	0.272
PEMS08	0.293	0.265	0.346	0.404	0.210	0.214	0.309	0.378	0.147	0.185	0.239	0.326

Table 15. Full results of anomaly detection on UCR Anomaly Archive, which contains 250 datasets (arranged in 25 rows for a total of 10 rows). We provide the quantile (%) of each dataset, where the bold parts represent the better results that benefited from pre-training.

Index		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	**22**	23	24	25
Timer (From Scratch)	1	1.1	16.2	6.7	1.2	19.0	23.8	16.7	19.0	14.3	2.4	0.5	13.3	2.0	1.8	0.0	0.4	0.4	30.7	3.0	3.0	7.6	1.3	3.0	47.6	14.3
	2	1.7	81.7	28.3	25.0	2.4	2.1	1.7	3.3	1.7	4.2	13.3	0.4	8.1	8.9	28.5	7.7	89.7	0.7	0.9	22.8	15.3	4.2	2.6	2.6	9.0
	3	5.1	1.3	62.5	2.8	9.5	19.9	71.8	16.0	5.0	2.7	7.4	12.8	4.1	1.2	53.0	3.3	33.3	94.0	1.5	0.3	0.3	20.6	1.0	30.0	16.3
	4	0.1	24.3	18.9	11.7	21.9	92.6	25.3	18.7	0.3	5.2	0.2	3.4	1.3	11.1	15.1	16.3	10.5	0.4	1.2	23.8	0.4	3.0	95.0	1.7	0.4
	5	32.5	0.7	1.3	62.7	21.7	1.3	43.8	36.5	0.6	9.1	3.5	1.2	19.0	21.4	14.3	23.8	16.7	2.4	0.5	14.0	2.0	1.3	0.0	0.4	0.4
	6	22.8	4.5	3.0	3.0	1.3	3.0	40.5	4.8	1.7	96.7	30.0	27.1	2.4	2.1	1.7	3.3	1.7	6.2	10.0	0.4	9.7	9.3	29.7	0.9	91.0
	7	0.7	1.4	23.5	11.1	4.2	2.6	1.3	17.9	5.1	1.3	68.8	2.8	9.5	15.8	59.1	11.9	2.6	0.5	16.4	20.7	1.3	0.5	36.4	3.3	37.5
	8	96.4	1.5	0.3	0.3	15.0	0.7	30.3	23.1	0.1	17.9	16.7	29.3	28.6	57.1	14.8	21.3	0.3	4.6	0.2	3.2	1.6	15.0	14.1	14.8	9.4
	9	0.3	21.3	79.6	2.9	14.1	4.0	1.6	6.0	0.1	49.3	31.7	30.1	0.0	0.0	0.1	0.0	0.2	0.0	0.0	0.0	0.0	0.1	17.5	13.3	7.3
	10	0.1	10.4	1.0	10.0	18.6	16.1	9.6	0.9	40.1	0.7	24.4	0.7	2.5	2.4	9.8	1.1	7.4	7.6	1.2	0.9	4.4	10.9	19.7	53.8	30.6
Timer (Pre-trained)	1	3.9	**5.8**	**1.5**	**1.2**	23.8	**19.0**	**7.1**	47.6	**11.9**	**2.4**	**0.5**	**6.7**	**2.0**	32.9	**0.0**	**0.4**	**0.4**	**26.8**	**3.0**	**3.0**	**6.1**	**1.3**	**1.5**	**2.4**	**4.8**
	2	**1.7**	**10.0**	**3.3**	**6.2**	4.8	**2.1**	**1.7**	**3.3**	**1.7**	**2.1**	**6.7**	**0.4**	**2.3**	**4.7**	28.9	**3.4**	**89.3**	**0.7**	1.4	**21.0**	**9.0**	**3.3**	**2.6**	**2.6**	**7.7**
	3	**5.1**	**1.3**	72.9	**2.8**	**2.4**	36.5	**45.5**	22.4	6.3	3.4	9.8	18.8	7.7	1.9	63.6	8.3	56.2	**77.4**	**1.5**	**0.3**	**0.3**	24.7	**0.7**	**15.8**	**8.7**
	4	0.6	**0.3**	19.0	14.3	66.9	92.9	27.8	**14.3**	**0.3**	**3.9**	**0.2**	5.9	6.0	17.0	**5.6**	**13.9**	**5.3**	0.9	**1.2**	38.1	**0.4**	**1.5**	**18.3**	**1.7**	**0.4**
	5	**31.7**	**0.7**	**1.3**	**26.4**	**12.4**	1.9	54.2	77.5	**0.6**	**1.5**	**0.6**	**1.2**	**16.7**	**16.7**	**4.8**	35.7	**9.5**	**2.4**	**0.5**	**7.3**	**2.0**	29.4	**0.0**	**0.4**	**0.4**
	6	**20.6**	**3.0**	**3.0**	**1.5**	**1.3**	**1.5**	**2.4**	**2.4**	**1.7**	**18.3**	**1.7**	**6.2**	**2.4**	**2.1**	**1.7**	**3.3**	**1.7**	**2.1**	**6.7**	**0.4**	**2.7**	**4.3**	**28.5**	1.3	**90.6**
	7	**0.7**	**0.9**	**21.0**	**9.0**	**3.3**	**1.3**	**1.3**	**3.8**	**5.1**	**1.3**	85.4	**2.8**	**3.1**	28.9	**25.5**	20.7	3.0	4.5	**13.3**	**15.8**	2.6	1.2	47.0	**3.3**	47.9
	8	**47.6**	**1.5**	**0.3**	**0.3**	19.7	1.0	**13.3**	**7.2**	0.1	**0.1**	**15.6**	96.5	96.3	74.1	**13.7**	**19.0**	**0.3**	**3.3**	**0.2**	5.4	8.9	**13.4**	**14.1**	**14.5**	**4.2**
	9	**0.3**	**1.2**	**63.5**	**0.3**	19.2	**1.7**	**0.3**	18.6	**0.1**	**29.8**	38.9	**16.2**	**0.0**	**0.0**	**0.0**	**0.0**	6.5	**0.0**	**0.0**	**0.0**	**0.0**	**0.1**	**16.5**	**11.0**	**5.7**
	10	**0.1**	**9.2**	**0.6**	**8.0**	**6.9**	**9.8**	**1.7**	1.2	**38.6**	1.1	**23.3**	**0.7**	**1.5**	**2.3**	10.2	1.1	8.5	**7.4**	**0.7**	**0.6**	5.5	17.0	28.8	**39.7**	**9.7**

Table 16. Detailed results for scaling up the pre-trained scale and the parameter of Timer.

Pre-trained		4G				4G			1G	2G	4G	12G
Model Dim.		256				512	768	1024	1024			
Layers		2	4	6	8	6			8			
5% Samples	PEMS03	0.188	0.174	0.168	**0.160**	0.146	0.138	**0.133**	0.130	0.128	0.128	**0.125**
	PEMS04	0.223	0.208	0.200	**0.190**	0.166	0.154	**0.145**	0.145	0.142	0.143	**0.135**
	PEMS07	0.147	0.131	0.123	**0.120**	0.106	0.097	**0.092**	0.092	0.090	0.090	**0.087**
	PEMS08	0.367	0.339	0.322	**0.319**	0.289	0.256	**0.239**	0.228	0.221	0.216	**0.204**
20% Samples	PEMS03	0.154	0.141	0.137	**0.134**	0.127	0.124	**0.123**	0.121	0.120	0.117	**0.114**
	PEMS04	0.182	0.162	0.155	**0.150**	0.140	0.132	**0.124**	0.124	0.123	0.122	**0.115**
	PEMS07	0.115	0.104	0.098	**0.095**	0.086	0.082	**0.080**	0.079	0.079	0.078	**0.076**
	PEMS08	0.326	0.277	0.247	**0.238**	0.206	**0.193**	0.194	0.193	0.185	**0.185**	0.187

Table 17. Additional downstream 672-pred-96 forecasting results of the subsets of PEMS and ETT under different data scarcity of the encoder-only and decoder-only Transformer. The bold part indicates that the result performs best in the current dataset and sample ratio.

Scenario	1% Target				5% Target				20% Target			
Architecture	Encoder		Decoder		Encoder		Decoder		Encoder		Decoder	
Pre-trained	None	12G	None	12G	None	12G	None	12G	None	12G	None	12G
ETTh1	0.446	0.413	0.428	**0.366**	0.437	0.405	0.426	**0.362**	0.409	0.404	0.385	**0.359**
ETTh2	0.338	0.304	0.315	**0.284**	0.329	0.293	0.314	**0.280**	0.308	0.299	0.294	**0.284**
ETTm1	0.463	0.370	0.407	**0.345**	0.391	0.340	0.354	**0.321**	0.344	0.323	0.332	**0.321**
ETTm2	0.220	**0.181**	0.207	0.183	0.197	**0.174**	0.190	0.176	**0.177**	0.179	**0.177**	0.187
PEMS03	0.225	0.196	0.249	**0.151**	0.165	0.160	0.158	**0.125**	0.144	0.145	0.135	**0.116**
PEMS04	0.253	0.226	0.320	**0.172**	0.198	0.184	0.195	**0.135**	0.167	0.161	0.145	**0.120**
PEMS07	0.170	0.156	0.179	**0.112**	0.126	0.125	0.114	**0.087**	0.102	0.103	0.093	**0.077**
PEMS08	0.496	0.405	0.563	**0.286**	0.389	0.319	0.391	**0.204**	0.280	0.246	0.241	**0.193**

Table 18. Zero-shot forecasting evaluation. We extensively evaluate available large time series models. We provided the average rank on all downstream datasets, where the lower is better. For the probabilistic forecaster Chronos: S1 means sampling one trajectory and S20 means sampling 20 trajectories and using the average. '-' indicates that multi-step error accumulation leads to failure predictions.

MODEL	TIMER-1B	TIMER-16B	TIMER-28B	MOIRAI-S	MOIRAI-M	MOIRAI-L	MOMENT	TIMESFM	CHRONOS-S1	CHRONOS-S20
ETTH1	0.438	0.364	0.393	0.441	0.383	0.394	0.674	0.414	0.571	0.454
ETTH2	0.314	0.294	0.308	0.295	0.295	0.293	0.330	0.318	0.423	0.326
ETTM1	0.690	0.766	0.420	0.562	0.448	0.452	0.670	0.354	0.632	0.451
ETTM2	0.213	0.234	0.247	0.218	0.225	0.214	0.257	0.201	0.272	0.190
ECL	0.192	0.139	0.147	0.212	0.162	0.155	0.744	-	-	-
TRAFFIC	0.458	0.399	0.414	0.616	0.425	0.399	1.293	-	-	-
WEATHER	0.181	0.203	0.243	0.195	0.197	0.221	0.255	-	-	-
RANK (AVG.)	**1.571**			2.286			4.429	2.250	5.500	3.250

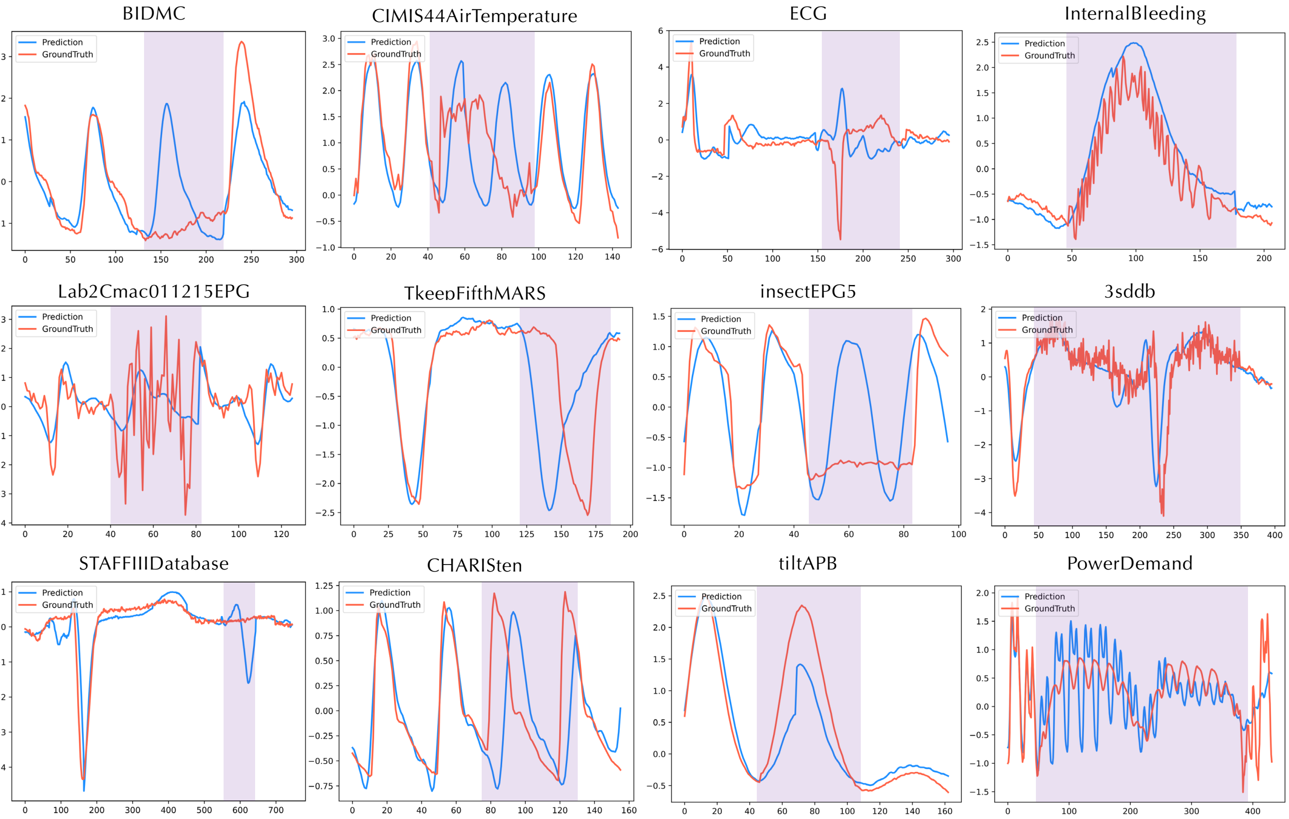

Figure 20. Visualization of anomaly detection results of Timer on partial UCR Anomaly Archive (Wu & Keogh, 2021). The masked part represents the abnormal position, and the model locates the abnormal interval by generating results that deviate from the abnormal series.

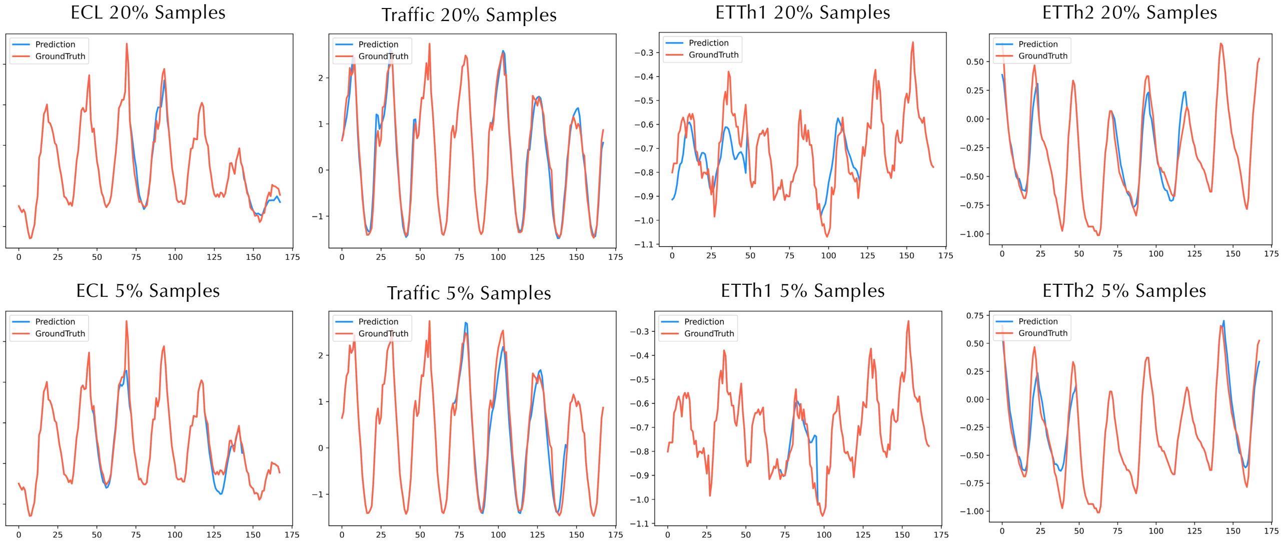

Figure 21. Visualization of imputation results of Timer trained with 5% and 20% samples.