# NuwaTS: a Foundation Model Mending Every Incomplete Time Series

Jinguo Cheng<sup>1</sup>, Chunwei Yang<sup>1</sup>, Wanlin Cai<sup>1</sup>, Yuxuan Liang<sup>2</sup> and Yuankai Wu<sup>1†\*</sup>
<sup>1</sup>Sichuan University. <sup>2</sup>The Hong Kong University of Science and Technology (Guangzhou).

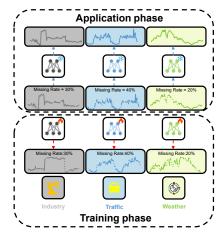
### **Abstract**

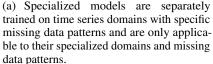
Time series imputation plays a crucial role in various real-world systems and has been extensively explored. Models for time series imputation often require specialization, necessitating distinct designs for different domains and missing patterns. In this study, we introduce NuwaTS, a framework to repurpose Pre-trained Language Model (PLM) for general time series imputation. Once trained, this model can be applied to imputation tasks on incomplete time series from any domain with any missing patterns. We begin by devising specific embeddings for each sub-series patch of the incomplete time series. These embeddings encapsulate information about the patch itself, the missing data patterns within the patch, and the patch's statistical characteristics. To enhance the model's adaptability to different missing patterns, we propose a contrastive learning approach to make representations of the same patch more similar across different missing patterns. By combining this contrastive loss with the missing data imputation task, we train PLMs to obtain a one-for-all imputation model. Furthermore, we utilize a plug-and-play layer-wise fine-tuning approach to train domain-specific models. Experimental results demonstrate that leveraging a dataset of over seventeen million time series from diverse domains, we obtain a one-for-all imputation model which outperforms existing domain-specific models across various datasets and missing patterns. Additionally, we find that NuwaTS can be generalized to other time series tasks such as forecasting.

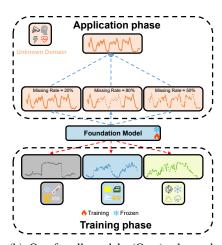
# 1 Introduction

Time series data are pervasive across numerous fields, including transportation [16], economics, healthcare [34], and meteorology [2]. However, real-world time series data often suffer from missing values. Incomplete data complicate various time series applications such as forecasting and classification, ultimately degrading the quality of data analysis [24]. This makes time series imputation especially critical. Traditionally, time series imputation methods have leaned heavily on statistical techniques like mean imputation and interpolation [36]. Yet, with the rise of deep learning, there has been a notable shift towards deep learning models for imputation tasks [6; 8]. Despite significant advancements in deep learning-based imputation methods, they often encounter limitations as they are generally trained and utilized within a single domain-specific dataset. For instance, a model trained on time series data from the transportation sector may not be effective for imputing missing data in other domains such as weather and electricity (Figure 1(a)).

In contrast, foundation models from the fields of Computer Vision (CV) [15] and Natural Language Processing (NLP) [4; 29], pre-trained on vast and diverse datasets from various domains, have shown remarkable versatility. These models can be adapted to a wide array of applications across different domains through methods such as fine-tuning [10; 45] or prompting [11; 15]. Inspired by these achievements, we aim to develop a versatile one-for-all model that can impute incomplete time series data from any domain and accommodate any pattern of missing data. (Figure 1(b)).







(b) One-for-all models (Ours): learned from relatively complete time series across various domains. Once trained, they can be applied to impute any incomplete time series from any domain.

Figure 1: Comparison between one-for-all models and specialized models for missing data imputation on time series.

However, effectively learning a one-for-all imputation model for diverse domains is technically challenging. The model must possess strong adaptability to various domains and missing data patterns. Additionally, in some applications, it needs the capability to quickly specialize to a specific domain with few-shot learning while retaining its generalizability. To tackle these challenges, we have developed NuwaTS, a one-for-all model for incomplete time series, distinguished by several strategies: First, recognizing that patches based on sub-series offer richer semantic information than individual time points, we have designed specific embeddings for each sub-series patch. These tokens include information for the patch itself, the missing data patterns within the patch, and the patch's statistical information, serving as inputs to the PLMs. Second, we introduced a contrastive learning module that encourages the model to produce more similar representations of the same patch under varying missing data patterns. The contrastive loss and the reconstruction loss together constitute the loss function we use to train the PLM parameters. Third, when a specialized model tailored to a specific domain is required, we have designed a domain-specific prefix embedding and developed a plug-and-play fine-tuning mechanism. This mechanism introduces modules that insert well-designed continuous prompts into each layer of the frozen pre-trained one-for-all model without altering any of its weights. Once these modules are removed, the model reverts to its original state as a versatile, one-for-all model.

Our contributions are as follows:

- 1. We introduce a novel solution, NuwaTS, capable of performing missing data imputation tasks on any incomplete time series unseen during the training phase. To the best of our knowledge, this represents the pioneering attempt in this field.
- 2. We present a "plug-and-play" fine-tuning technique that seamlessly transforms a one-for-all model into a domain-specific model with minimal data and computational resources, all without modifying the model's original weights.
- 3. The one-for-all NuwaTS consistently outperforms domain-specific state-of-the-art methods in imputation tasks across nearly all missing rates. Moreover, fine-tuned NuwaTS can be extended to time series forecasting, where its forecasting results are comparable to or even better than existing domain-specific time series forecasting models.

# 2 Related Works

# 2.1 Incomplete Time Series Imputation

Many time series imputation models are tailored to specific missing data patterns and domains, such as randomly missing traffic time series. For instance, matrix factorization models are designed for multivariate time series imputation [40]. For a dataset from a specific domain with a particular missing rate, it often requires optimizing a specialized model using a low-rank prior as the optimization target for imputation. Recent advancements in deep learning techniques have shown promising results in addressing missing data imputation. Generative models such as Generative Adversarial Networks (GANs)[23; 33] and diffusion models are used to learn the underlying distribution of the incomplete time series. Several architectures, such as Recurrent Neural Networks (RNNs)[6; 25; 22; 32] and attention mechanisms [8], are proposed to capture temporal dependencies within incomplete time series. While achieving good performance on narrow domains and missing data patterns, these models lack versatility and generalizability to diverse domains and missing data patterns.

### 2.2 Foundation Model for Time Series

In recent years, foundational models for time series analysis have made notable strides, primarily leveraging pre-trained backbones from NLP and aligning modalities to extend their reasoning capabilities to time series tasks. For example, Gruver et al. [9] discovered that by encoding time series as strings of numerical digits, LLMs can perform time series forecasting with zero-shot capabilities. GPT4TS [44] trains time series models using pre-trained GPT weights. UniTime [18] integrates domain-specific instructions into LLMs, enhancing their generalization across domains. TIME-LLM [12] employs LLMs' reasoning by framing time series data and statistical information in textual prompts, aligning time patches with word embeddings via cross-attention mechanisms for improved zero-shot learning. Autotimes [21] utilizes precomputed text embedding as positional embeddings and an autoregressive approach for long-term forecasting. TEST [31] uses text-prototypealigned embeddings to enhance LLMs' reasoning in time series data. S<sup>2</sup>IP-LLM [28] applies seasonal-trend decomposition to time series and uses semantic space-informed prompting to retrieve appropriate prompts from word token embeddings as prefixes. LLM4TS [3] adapts LLMs for time series representation learning by directly converting individual patches into time series sequences. Chronos [1] trains language models from scratch on a large collection of time series data, using scaling and quantization for tokenization. This model demonstrates strong capabilities in zeroshot probabilistic forecasting. These models primarily focus on time series forecasting and do not specifically address missing data issues or the design of embeddings for missing data patterns.

# 3 Methodology

### 3.1 Preliminaries and Problem Statement

**Definition 1** Incomplete Time Series: We define an incomplete time series  $\mathbf{x} = \{x_1, x_2, \dots, x_T\} \in \mathbb{R}^T$  from domain S as a sequence of T observations. Each observation  $x_t$  is associated with a timestamp  $s_t$ . In practice, an observation  $x_t$  may not be observable due to various reasons. To represent the missing values in  $\mathbf{x}$ , we introduce a masking vector  $\mathbf{m} \in \mathbb{R}^T$  defined by:

$$m_t = \begin{cases} 1 & \text{if } x_t \text{ is observed} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Suppose we have acquired N relatively complete time series datasets  $\mathcal{D} = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N\}$  from a diverse set of domains  $\{\mathcal{S}^1, \mathcal{S}^2, \dots, \mathcal{S}^K\}$ , where  $\mathbf{x}^n \in \mathbb{R}^{T_n}$  represents the  $n^{th}$  time series with length  $T_n$ . Here, a "relatively complete time series" implies that  $\sum m_t^n \ll T_n$ , where  $\mathbf{m}^n$  denotes the missing data pattern in the n-th series. Our objective is to utilize  $\mathcal{D}$  to train an imputation model, denoted as  $f_{\mathbf{\Phi}}$ , characterized by parameters  $\mathbf{\Phi}$ . For any incomplete time series  $\hat{\mathbf{x}} \in \mathbb{R}^{\hat{T}}$  accompanied by any missing pattern  $\hat{\mathbf{m}} \in \mathbb{R}^{\hat{T}}$  from any domain  $\hat{\mathcal{S}}$ , the model  $f_{\mathbf{\Phi}}$  can impute the missing values in  $\hat{\mathbf{x}}$  as accurately as possible.

### 3.2 Model Architecture

In this work, we initialize model parameters  $\Phi$  with weights from PLMs. Given the constraints of computational resources, we limit our use to the parameters of the first six layers, thereby making NuwaTS more accessible to applications with limited computational resources.

To train PLMs to adapt to various domains and missing data patterns, we have implemented several key design features for training. These include Instance Normalization & Patching, Statistical Embedding, Missing Embedding, a Domain-Specific Embedding for fine-tuning and Contrastive Learning with Variable Missing Patterns. We visualize the model architecture in Figure 2.

# 3.2.1 Instance Normalization & Patching

Time series from different domains may exhibit variations in magnitude and distribution. To mitigate these discrepancies, we apply reversible instance normalization [14] to each variable  $\mathbf{x}_i \in \mathbb{R}^L$  (where  $\mathbf{x}_i$  is a segment of length L randomly extracted from a time series in the dataset  $\mathcal{D}$ ) before inputting it into the model. For the missing values, we assign them a value of 0. Recognizing that individual time points lack sufficient semantic meaning and hinder the model's ability to recognize domain information, we utilize patching techniques. We define the patch length as P and the non-overlapping region between two consecutive patches as S. The patching process then generates a sequence of patches  $\mathbf{X}_{i,(p)} \in \mathbb{R}^{P \times N}$ , where N is the number of patches, calculated as  $N = \left| \frac{L-P}{S} \right| + 2$ . Additionally, we pad the sequence with S re-

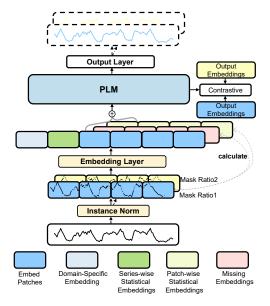


Figure 2: Overview of NuwaTS. To fully leverage the semantic information of time series and their missing patterns, NuwaTS introduces the tokenization of time series in patches. It utilizes the missing data patterns, statistical information for each pattern and the entire series, and a domain-specific embedding, trained through imputation and contrastive learning tasks.

peated values of the last data point  $\mathbf{x}_i$  before initiating the patching process. We then use a shared and learnable linear projection to embed each patch into a hidden space  $\mathbf{Z}_{i,(p)} \in \mathbb{R}^{D \times N}$ , where D is set to match the dimension used in the subsequent model.

# 3.2.2 Statistical Embedding

Previous work primarily utilized hard textual prompts to translate dataset descriptions and statistical information [18; 21]. Due to the fact that time series patches and textual prompts come from two different sources, aligning them has proven to be a complex task [12]. In this work, we move away from using hard textual prompts to describe time series statistics. Instead, we generate crucial statistical information separately for both the entire segment and each non-overlapping patch, including the minimum, median, and maximum values (after normalization), as well as trends (calculated as the last observation value of a patch minus its first). This information is represented as  $\mathbf{V}_i \in \mathbb{R}^{4 \times (N+1)}$ . Similarly, we employ a shared and learnable linear projection to embed  $\mathbf{V}_i$  into a hidden space, denoted by  $\mathbf{Z}_{i,(v)} \in \mathbb{R}^{D \times (N+1)}$ , where  $\mathbf{Z}_{i,(v)} = [\mathbf{z}_{i,(v_g)}, \mathbf{Z}_{i,(v_p)}]$ , within this embedding, the first embedding  $\mathbf{z}_{i,(v_g)} \in \mathbb{R}^D$  represents the statistical information for the entire time series. The remaining  $\mathbf{Z}_{i,(v_p)} \in \mathbb{R}^{D \times N}$  pertains to each individual patch.

# 3.2.3 Missing Embedding

Adapting to the missing data pattern is also a crucial aspect. The missing rate of each patch accurately reflects the distribution of missing data across the target time series. Therefore, we have also designed a Missing Embedding  $\mathbf{z}_{i,(m)} \in \mathbb{R}^{D \times 1}$ . It is a randomly-initialized and learnable parameter. We

multiply it with the corresponding patch's mask ratio  $\mathbf{r}_i \in \mathbb{R}^{1 \times N}$  and add it to the final embedding of the patching information:  $\mathbf{E}_{i,(p)} = \mathbf{Z}_{i,(p)} + \mathbf{Z}_{i,(v_n)} + \mathbf{z}_{i,(m)} \times \mathbf{r}_i$ .

# 3.2.4 Domain-Specific Embedding

In some instances, we may need our one-for-all model to function within a specific domain for which we have training data, and we also want to preserve its capability for cross-domain generalization. To achieve this, we explicitly introduce a new embedding,  $\mathbf{k} \in \mathbb{R}^D$ . We anticipate that this embedding will encapsulate unique information from the training data throughout the training process and serve as a repository of domain-specific knowledge. We insert this embedding before the patch embeddings  $\mathbf{E}_{i,(p)}$ . By integrating the domain-specific embedding and the global statistical embedding with patch embeddings  $\mathbf{E}_{i,(p)}$ , the final embeddings for input  $\mathbf{x}_i$  is expressed as  $\mathbf{E}_i = [\mathbf{k}, \mathbf{z}_{i,(v_o)}, \mathbf{E}_{i,(p)}]$ , where  $\mathbf{E}_i \in \mathbb{R}^{(N+2)\times D}$ .

# 3.2.5 Output Layer

After passing through the PLM backbone, we discard the segments corresponding to the two added prefixes: the domain-specific embedding and the variable-specific embedding. This means that we use only the representations of the N patches corresponding to the time series as inputs for the model's final layer. Although the first two embeddings also participate in the attention calculations, they are not included in the final input. We then flatten the final representation to obtain  $\mathbf{h}_i^{(\mathrm{Layer})} \in \mathbb{R}^{N \times D}$ . Finally, we linearly map these representations back to their original dimensions, obtaining the model outputs  $\mathbf{o}_i \in \mathbb{R}^L$ .

# Missing Rate pos neg Series 0 Missing Rate 20% Missing Rate 20% Series 0 Series 0 Series 1

Figure 3: Illustration of contrastive learning: Representations of each patch under different masks are treated as positive samples for each other, while representations from other time series and different patches from the same sequence under varying masks are considered negative samples.

# 3.2.6 One-For-All Model Training

To further enhance our model's adaptability to different missing patterns, we have incorporated a Contrastive Learning module into our training process. The principle behind this module is straightforward: it ensures that the model can learn relatively similar representations of the same patch, when subjected to various missing rates and patterns. During training, for each input  $\mathbf{x}_i \in \mathbb{R}^L$ , we generate two random masks,  $\mathbf{m}_{i,1}$  and  $\mathbf{m}_{i,2} \in \mathbb{R}^L$ , to mask the inputs. These two masked time series are then simultaneously used as inputs for the PLM. Each patch outputs a representation and an output head is utilized for contrastive learning. As illustrated in Figure 3, we treat the representations of each patch under different masks as positive samples for each other, while representations from other time series and other patches from the same series under different masks are treated as negative samples. To filter out some similar samples, we do not use representations from other patches of the same time series under the same masks as negative samples. We use the InfoNCE loss [27] combined with the Mean Squared Error (MSE) loss between the model outputs and the original time series as the loss function for the one-for-all model (see Appendix B.3 for details). For training parameters, we freeze the self-attention layers of the PLM and only train the layer normalization, word positional embeddings, and feed-forward neural networks within the PLM.

# 3.3 Domain-Specific Fine-Tuning

To achieve a domain-specific model, we borrow the idea of P-tuningv2 strategy [17]. Unlike pre-training phase, where the domain-specific embedding  $\mathbf k$  is only added to the input embedding. We incorporate this domain-specific information into every layer of the PLM. Initially, we employ a domain-transfer layer—a two-layer MLP network—to map  $\mathbf k \in \mathbb{R}^D$  to  $\hat{\mathcal K} \in \mathbb{R}^{2 \times \text{Layer} \times D}$ . Following the P-tuningv2 approach, we also randomly initialize a continuous prompt  $\mathcal P \in \mathbb{R}^{2 \times \text{Layer} \times D}$  for each layer. We then combine the randomly initialized  $\mathcal P$  with the domain-specific information  $\hat{\mathcal K}$  to serve as the domain-specific prefix Key and Value in the PLM's hidden layers. Thus, we have  $[\mathbf{Key}_p, \mathbf{Value}_p] = \mathcal P + \beta \hat{\mathcal K}$ , where  $\beta = 0.01$ .

Here,  $\mathbf{Key}_p$  and  $\mathbf{Value}_p \in \mathbb{R}^{\mathrm{Layer} \times D}$  contain the domain-specific prefix key and value for each layer and participate in the subsequent attention computation. During fine-tuning on time series data from the target domain, we freeze all parameters except for the randomly initialized  $\mathcal P$  and the domain-transfer layer. The fine-tuning process is illustrated in Figure 4. If these two trainable modules are removed, our domain-specific model reverts to the original one-for-all foundation model. Therefore, the fine-tuning strategy we propose is essentially "plug-and-play".

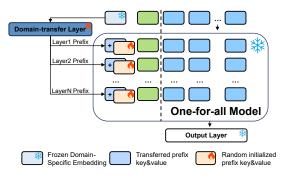


Figure 4: Illustration of domain-specific fine-tuning.

# 4 Experiment

We conducted comprehensive experiments on NuwaTS across ten commonly used datasets from various domains. To facilitate comparison, we provided three versions of the model based on the GPT2 architecture: a **specific model** trained on a single domain, an **one-for-all model** trained on a general fused dataset (17.6M collected samples from various domains), and a **fine-tuned model** for a specific domain (fine-tuned based on the one-for-all model). Moreover, we evaluated BERT [7] and LLaMA2 [35] as backbones for NuwaTS (details in Appendix D.3). To assess its zero-shot capability, we trained another GPT2-based model on a transportation domain dataset, LargeST [19] (comprising 100.1 million collected samples) and evaluate it on time series from other domains.

**Baselines.** Following the configuration outlined in [8], we evaluated two naive imputation methods: **Median**, where missing values are filled with the median value, and **Last**, where missing values are filled with the last previous observations. To ensure a fair comparison, all methods followed a unified pipeline<sup>2</sup>. We assessed two classic deep learning-based imputation models, **BRITS**[6] and **SAITS**[8]. We also compared our model with other foundation models for time series such as **DLinear**[41], **PatchTST**[26], **iTransformer**[20], **TimesNet**[38], **Autoformer**[39], **Fedformer**[43], and **GPT4TS**[44]. We trained GPT4TS separately on single datasets with the number of layers set to 6 to align with NuwaTS. We trained all these models using a domain-specific approach. Since PatchTST is a transformer-based model with bi-directional attention, we also trained and tested it on the general fused dataset.

**Setups.** We partitioned the dataset along the sensor (variable) dimension into training, validation, and test sets in a 1:1:1 ratio. This division simulates the process of collecting relatively complete time series data from a few sensors or variables with lower missing rates, training a model with this data, and then using it to impute data from other sensors or variables that have higher missing rates. The input time series length L is 96, and we trained all baselines under random missing rates (sampled from 0.1 to 0.9) and tested them separately under 9 missing rates: 0.1, 0.2,..., 0.9. We used a total of ten datasets for domain-specific training and mixed them into a general fused dataset for one-for-all training. Appendix A.1 shows the details of datasets. For zero-shot training, we chose LargeST $^3$ , a large-scale traffic dataset including a total of 8,600 time series and over 100 million samples, to train NuwaTS and conduct zero-shot experiments on other domains such as ETTs, weather and electricity.

### 4.1 Main Results

We present the average MAE and MSE for specific, one-for-all, and fine-tuned NuwaTS models, along with other baselines, across 10 different datasets in Table 1 and Table 2. Furthermore, we visualize the average MSE across all datasets for the NuwaTS models and several baselines in Figure 5. On nearly all datasets, the specific NuwaTS model outperforms other domain-specific models. The comparison with GPT4TS [44] is particularly noteworthy. GPT4TS also involves training on a PLM backbone. However, it not only underperforms compared to NuwaTS, but it is also worse than PatchTST. This underscores the importance of the patching technique<sup>4</sup> in helping the model

<sup>&</sup>lt;sup>2</sup>https://github.com/thuml/Time-Series-Library

<sup>3</sup>https://github.com/liuxu77/LargeST

<sup>&</sup>lt;sup>4</sup>GPT4TS employs a channel-dependent approach on imputation task.

Table 1: Imputation results are presented for ETTs, ECL, and weather datasets with nine test missing rates: 0.1, 0.2, ..., 0.9. All results are averaged across the nine different missing rates. A lower MSE or MAE signifies better imputation performance. **Red** highlights the best results, while **Blue** indicates the second-best.

Model Median Last	MSE 0.723 0.432	Th1 MAE 0.611 0.476	MSE 0.728 0.089	Th2 MAE 0.472 0.149	MSE 0.699 0.336	Tm1 MAE 0.584 0.413	MSE 0.744 0.059	Гm2 МАЕ 0.464 0.122	MSE 1.010 0.965	CL MAE 0.834 0.826	MSE 0.998 0.731	MAE 0.497 0.349
Autoformer(2021)	0.552	0.547	0.472	0.420	0.346	0.409	0.211	0.312	0.137	0.262	0.610	0.450
Fedformer(2022)	0.354	0.436	0.538	0.429	0.076	0.185	0.055	0.156	0.129	0.258	0.237	0.211
Dlinear(2023)	0.356	0.414	0.245	0.301	0.274	0.357	0.266	0.351	0.317	0.419	0.355	0.309
iTransformer(2024)	0.639	0.572	0.369	0.376	0.352	0.413	0.356	0.416	0.092	0.200	0.515	0.412
BRITS(2018) <sup>1</sup> TimesNet(2022) PatchTST(2023) SAITS(2023) <sup>1</sup> GPT4TS(2024) NuwaTS(specific)	0.213 0.166 0.185 0.196 0.196 0.182	0.313 0.280 0.298 0.298 0.290 0.290 0.293	0.117 0.021 0.022 0.215 0.025 0.020	0.172 0.091 0.093 0.190 0.092 0.091	0.096 0.066 0.080 0.087 0.078 0.070	0.183 0.155 0.183 0.163 0.161 0.164	0.071 0.011 0.011 0.135 0.012 0.011	0.123 0.063 0.066 0.139 0.063 0.064	0.317 0.362 0.136 0.445 0.296 0.086	0.427 0.450 0.266 0.512 0.401 0.191	0.793 0.681 0.271 0.932 0.939 0.307	0.474 0.280 0.122 0.484 0.321 0.127
PatchTST(one-for-all)	0.178	0.288	0.019	0.088	0.075	0.172	0.011	0.065	0.121	0.243	0.230	0.116
NuwaTS(one-for-all)	0.164	0.263	0.018	0.084	0.064	0.147	0.010	0.060	0.085	0.186	0.206	0.088
NuwaTS(fine-tuned)	0.156	0.255	0.017	0.082	0.060	0.142	0.010	0.058	0.081	0.183	0.207	0.086

<sup>&</sup>lt;sup>1</sup> We replace the MAE loss function in SAITS [8] and BRITS [6] with MSE loss function for fair comparison.

Table 2: Imputation results on four PEMS datasets with same setting as Table 1.

Model	PEN	IS03 MAE	PEN MSE	IS04 MAE	PEN MSE	IS07 MAE	PEN MSE	IS08 MAE
Median	0.691	0.609	0.742	0.645	0.755	0.646	0.734	0.653
Last	0.474	0.506	0.487	0.517	0.507	0.517	0.446	0.495
Autoformer(2021)	0.792	0.658	0.404	0.495	0.406	0.492	1.068	0.750
Fedformer(2022)	0.263	0.378	0.350	0.444	0.318	0.423	0.274	0.374
Dlinear(2023)	0.262	0.390	0.266	0.392	0.259	0.387	0.268	0.393
iTransformer(2024)	0.108	0.237	0.131	0.259	0.099	0.225	0.162	0.291
BRITS(2018)	0.143	0.267	0.259	0.370	0.228	0.353	0.225	0.344
TimesNet(2022)	0.102	0.221	0.152	0.268	0.132	0.252	0.114	0.231
PatchTST(2023)	0.059	0.170	0.074	0.190	0.052	0.159	0.067	0.180
SAITS(2023)	0.157	0.286	0.271	0.380	0.228	0.349	0.250	0.364
GPT4TS(2024)	0.101	0.220	0.158	0.271	0.131	0.250	0.110	0.227
NuwaTS(specific)	0.049	0.149	0.058	0.159	0.040	0.129	0.057	0.156
PatchTST(one-for-all)	0.056	0.167	0.066	0.179	0.050	0.157	0.060	0.167
NuwaTS(one-for-all)	0.047	<u>0.146</u>	0.058	0.159	0.040	0.129	0.052	<u>0.146</u>
NuwaTS(fine-tuned)	0.047	<u>0.146</u>	0.058	0.159	0.040	0.129	0.052	<u>0.146</u>

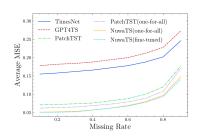


Figure 5: Main results across ten datasets under different missing rates.

understand missing data patterns. Moreover, we observed that the generalization ability of NuwaTS and PatchTST (one-for-all) was further augmented after training on the general fused dataset, thereby affirming the existence of a scaling law [13] within the realm of time series modeling. We visualized the imputation results in Appendix C.1.

# 4.2 Zero-shot Capability

We validated the zero-shot capability of our model by directly evaluating a trained model on target data without further training. TimesNet and GPT4TS, which use a channel-dependent approach, can only perform zero-shot across datasets with the same number of variables, specifically the four ETT datasets in our study. In contrast, NuwaTS and PatchTST, as channel-independent approach, support inference on datasets with different variable dimension. We trained them on LargeST and then evaluated them on ECL and Weather datasets. The results in Table 3 shows NuwaTS exhibits superior zero-shot capability, achieving better performance on all cases. The reason may lie in the use of specialized missing data embedding and statistical embedding. Additionally, the language knowledge embedded in the PLM has strong generalization capabilities, whereas the PatchTST model trained on LargeST does not possess such strong capabilities.

Table 3: Zero shot evaluation results. All methods were trained on one dataset and zero shot to the other. A lower MSE or MAE indicates a better imputation performance. **Red**: the best. /: model failed to work.

Metrics	MSE	waTS MAE	MSE	hTST MAE	MSE	T4TS MAE	MSE	esNet MAE
$ETTh1 \Rightarrow ETTh2 \\ ETTh1 \Rightarrow ETTm2 \\ ETTm1 \Rightarrow ETTm2 \\ ETTm1 \Rightarrow ETTm2 \\ ETTm1 \Rightarrow ETTm2 \\ LargeST \Rightarrow ECL \\ LargeST \Rightarrow Weather$	0.023 0.013 0.021 0.011 0.338 0.217	0.091 0.072 0.094 0.063 0.366 0.087	0.024 0.013 0.022 0.011 0.385 0.236	0.092 0.073 0.096 0.067 0.433 0.121	0.029 0.015 0.027 0.013	0.095 0.073 0.093 0.064 /	0.026 0.014 0.024 0.013	0.095 0.072 0.096 0.066 /

### 4.3 Few-shot Domain-Specific Fine-Tuning

The ability to generalize to the target domain with a small amount of data is an important criterion for evaluating the generality of a model. We use 10% and 1% of the data for fine-tuning the NuwaTS model trained on LargeST. The results shown in Table 4 indicate that domain-specific fine-tuning is particularly effective for fields with limited data. On the ETT datasets, we achieved the same results using only 10% of the data as we did using 100% of the data.

Table 4: Fine-tuned results from the model pre-trained on LargeST [19]. A lower MSE or MAE indicates a better imputation performance. **Red**: the best, **Blue**: the second best, **bold**↓: fine-tuning works.

Methods	ECL		ET	ETTh1		Th2	ET	Γm1	ETTm2	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Zero-Shot	0.338	0.366	0.212	0.306	0.021	0.089	0.066	0.145	0.011	0.059
Fine-tuning with 1% data	0.284↓	0.338↓	0.205↓	0.305↓	0.021↓	0.090	0.063↓	0.147	0.010↓	0.059
Fine-tuning with 10% data	0.190 ↓	0.292↓	0.180 ↓	0.282↓	0.019↓	0.086↓	0.060 ↓	0.142↓	0.010 ↓	0.058 ↓
Fine-tuning with 100% data	0.143 ↓	<b>0.253</b> ↓	<u>0.180</u> ↓	<b>0.280</b> ↓	<b>0.019</b> ↓	0.085	<u>0.061</u> ↓	<u>0.143</u> ↓	0.010	<u>0.058</u> ↓

### 4.4 Ablation Study

We conducted ablation experiments via zero-shot evaluation. We trained the default model, several ablated models on LargeST. We then performed zero-shot evaluation on ECL and Weather datasets for out-of-domain evaluation. As shown in Table 5, after ablation of the Statistic Embedding, Missing Embedding, and contrastive learning module, the model's zero-shot ability decreases significantly, proving the necessity of these key components. Freezing the PLM backbone leads to poor performance. Finally, we discovered that when training the model from scratch without using weights pretrained on NLP tasks, its zero-shot generalization

Table 5: Ablation results of the model training on LargeST and zero-shot on ECL and Weather. A lower MSE or MAE indicates a better imputation performance. **Red**: the best.

Model	MSE EG	CL MAE	Wea MSE	ther MAE
Default	0.338	0.366	0.217	0.087
w/o-StatisticEmbedding	0.371	0.383	0.224	0.093
w/o-ContrastiveLearning	0.348	0.366	0.218	0.087
w/o-MissingEmbedding	0.338	0.368	0.221	0.092
w/o-GPT2weight <sup>1</sup> FrozenBackbone	0.339 0.533	0.368 0.560	0.218 0.290	0.089 0.131

We trained NuwaTS from scratch without loading the pre-trained language weight.

ability on other datasets deteriorates significantly. This indicates that training the model on NLP tasks benefits its performance on time series tasks, demonstrating that cross-modality training is meaningful [42].

We also verified the design of domain-specific fine-tuning strategy of NuwaTS in Table 6. When the randomly initialized prefix key&value and the knowledge-transfer layer were ablated, the model's fine-tuning performance showed a significant decline which indicates that domain-transfer layer retains the learned knowledge while newly added prefix key&value provide flexibility in transferring to specific domain. Additionally, we find that merely appending the prefix to the input layer and fine-tuning it significantly reduces the effectiveness of fine-tuning. Inserting the prefix into each layer of the PLM enhances the prefix's representation capacity during domain transfer [17].

Table 6: Domain-specific fine-tuning ablation study on ETTs based on the model pre-trained on general fused dataset. A lower MSE or MAE indicates a better imputation performance. **Red**: the best.

Model	II ET	Th1	ET	Th2	ET	Γm1	ET	Γm2
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Domain-Specific Fine-Tuning	0.156	0.255	0.017	0.082	0.060	0.142	0.010	0.058
w/o random initialized prefix key&value	0.160	0.260	0.019	0.085	0.063	0.149	0.010	0.060
w/o domain-transfer layer	0.158	0.256	0.018	0.083	0.060	0.142	0.010	0.058
only apply to input embedding	0.163	0.263	0.018	0.084	0.062	0.145	0.010	0.060
no fine-tuning	0.164	0.263	0.018	0.084	0.064	0.147	0.010	0.060

# 4.5 Enhancing Forecasting on Incomplete Time Series

In practical applications, forecasting models often have to train on incomplete data, which can significantly impair their performance. To address this issue, we applied pre-trained imputation models to impute the incomplete data before training the forecasting model. We chose TimesNet [38]

Table 7: We trained and evaluated TimesNet [38] on incomplete time series with original missing rates of 0.2 and 0.5, comparing its forecasting performance on data imputed by PatchTST and NuwaTS. We set the length of input sequence and forecasting both to 96. A lower MSE or MAE indicates a better forecasting performance. **Red**: the best.

Method	is		MSE ET	Γh1 MAE	ET MSE	Th2 MAE	MSE ET	Γm1 MAE	ET MSE	Γm2 MAE
Complete	Data	Ш	0.384	0.402	0.340	0.374	0.338	0.375	0.187	0.267
Missing Rate: 0.2	Default +PatchTST +NuwaTS		0.457 0.439 <b>0.428</b>	0.450 0.445 <b>0.437</b>	0.353 0.346 <b>0.341</b>	0.389 0.382 <b>0.378</b>	0.389 0.353 <b>0.344</b>	0.404 0.388 <b>0.381</b>	0.200 <b>0.189</b> 0.190	0.284 0.270 <b>0.269</b>
Missing Rate: 0.5	Default +PatchTST +NuwaTS		0.585 0.450 <b>0.441</b>	0.515 0.452 <b>0.446</b>	0.370 0.354 <b>0.344</b>	0.403 0.389 <b>0.382</b>	0.572 0.357 <b>0.345</b>	0.496 0.391 <b>0.383</b>	0.238 0.189 0.190	0.318 0.270 0.270

as the forecasting model and used four ETT datasets for training. We randomly discarded 20% and 50% of the data in the training sets of four ETT datasets.

To prevent data leakage during imputation, we used NuwaTS and PatchTST pre-trained on LargeST to impute the training data. Finally, we trained TimesNet [38] on the imputed time series and tested it on the complete time series.

As shown in Table 7, the data imputed by NuwaTS improved the forecasting model's performance across most metrics. Additionally, the imputed data from NuwaTS was of higher quality for downstream tasks compared to PatchTST, demonstrating that NuwaTS can effectively address the issue of incomplete time series data in practical applications.

### 4.6 Fine-tuning Imputation Model to Forecasting Model

Table 8: Forecasting results. We set the length of input sequence and forecasting both to 96. Forecasting results from other baselines come from [38; 20; 5]. We fine-tuned the model pre-trained on LargeST in order to avoid data leakage. A lower MSE or MAE indicates a better performance. **Red**: the best, **Blue**: the second best.

Model	ETTh1		ETTh2		ETTm1		ET	Γm2	ECL		Wea	ther
	MSE	MAE										
Autoformer(2021)	0.449	0.459	0.346	0.388	0.505	0.475	0.255	0.339	0.201	0.317	0.266	0.336
Fedformer(2022)	0.395	0.424	0.358	0.397	0.379	0.419	0.203	0.287	0.193	0.308	0.217	0.296
TimesNet(2022)	0.384	0.402	0.340	0.374	0.338	0.375	0.187	0.267	0.168	0.272	0.172	0.220
Dlinear(2023)	0.397	0.412	0.333	0.387	0.345	0.372	0.193	0.292	0.197	0.282	0.196	0.255
PatchTST(2023)	0.460	0.447	0.308	0.355	0.352	0.374	0.183	0.270	0.190	0.296	0.186	0.227
MSGNet(2024)	0.390	0.411	0.328	0.371	0.319	0.366	0.177	0.262	0.165	0.274	0.163	0.212
iTransformer(2024)	0.386	0.405	0.297	0.349	0.334	0.368	0.180	0.264	0.148	0.240	0.174	0.214
NuwaTS(fine-tuned without output layer)	0.398	0.419	0.319	0.369	0.316	0.357	0.191	0.275	0.197	0.284	0.176	0.219
NuwaTS(fine-tuned with output layer)	0.375	0.404	0.292	0.348	0.319	0.360	0.185	0.268	0.183	0.267	0.172	0.215

By directly inserting the masked padding tokens  $\mathbf{p}$  where  $\mathbf{p} \in \mathbb{R}^{M \times D}$  after the original input embeddings  $\mathbf{E}_i \in \mathbb{R}^{(N+2) \times D}$ , we get  $\mathbf{E}_i^f = [\mathbf{k}, \mathbf{z}_{i,(v_g)}, \mathbf{E}_{i,(p)}, \mathbf{p}]$  where  $\mathbf{E}_i^f \in \mathbb{R}^{(N+2+M) \times D}$ . The model automatically imputes the future M patches, effectively transforming into a forecasting model. We discard the prefixal part and obtain the final forecasting representation  $\mathbf{h}_i^{(\text{Layer})} \in \mathbb{R}^{M \times D}$ . Then  $\mathbf{h}_i^{(\text{Layer})}$  pass through the output layer and we get the final forecasting results. In order to further help NuwaTS adapt to the forecasting task, we applied domain-specific fine-tuning on the one-for-all model and trained its domain-transfer layer, LayerNorm layers as well as position embeddings, which account for only 9.35% of the model's total parameters (details in Appendix A.3). Notably, even without training the output layer, the model still achieves remarkable performance on the long-term forecasting task, as shown in Table 8. We visualized the forecasting results in Appendix C.2.

# 5 Conclusion

We propose NuwaTS, a time series imputation model equipped with a PLM backbone and a contrastive learning module that explicitly incorporates patch-wise statistical information and missing patterns. Experimental results demonstrate that our imputation model significantly outperforms existing methods and can adapt to diverse domains and missing rates in a zero-shot manner. Additionally, we developed an efficient domain-specific fine-tuning technique that, with minimal cost, can adapt the model from an all-in-one to a domain-specific one, and also transform it from an imputation model to a forecasting model. To the best of our knowledge, this is the first attempt to design a foundational

time series imputation model that can be applied across different domains. Our proposed method serves as a valuable reference for this field.

### References

- [1] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.
- [2] Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.
- [3] Yuxuan Bian, Xu Ju, Jiangtong Li, Zhijian Xu, Dawei Cheng, and Qiang Xu. Multi-patch prediction: Adapting Ilms for time series representation learning. *ArXiv*, abs/2402.04852, 2024.
- [4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [5] Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 11141–11149, 2024.
- [6] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. Brits: Bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31, 2018.
- [7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [8] Wenjie Du, David Côté, and Yan Liu. Saits: Self-attention-based imputation for time series. Expert Systems with Applications, 219:119619, 2023.
- [9] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. Large language models are zero-shot time series forecasters. Advances in Neural Information Processing Systems, 36, 2024.
- [10] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*, 2022.
- [11] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022.
- [12] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *The Twelfth International Conference on Learning Representations*, 2024.
- [13] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361, 2020.
- [14] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.

- [15] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 4015–4026, 2023.
- [16] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.
- [17] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland, May 2022. Association for Computational Linguistics.
- [18] Xu Liu, Junfeng Hu, Yuan Li, Shizhe Diao, Yuxuan Liang, Bryan Hooi, and Roger Zimmermann. Unitime: A language-empowered unified model for cross-domain time series forecasting. *CoRR*, abs/2310.09751, 2023.
- [19] Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, Lei Bai, Chao Huang, Zhenguang Liu, Bryan Hooi, and Roger Zimmermann. Largest: A benchmark dataset for large-scale traffic forecasting. *Advances in Neural Information Processing Systems*, 36, 2024.
- [20] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.
- [21] Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Autotimes: Autoregressive time series forecasters via large language models. arXiv preprint arXiv:2402.02370, 2024.
- [22] Yukai Liu, Rose Yu, Stephan Zheng, Eric Zhan, and Yisong Yue. Naomi: Non-autoregressive multiresolution sequence imputation. Advances in neural information processing systems, 32, 2019.
- [23] Yonghong Luo, Ying Zhang, Xiangrui Cai, and Xiaojie Yuan. E<sup>2</sup>gan: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 3094–3100. International Joint Conferences on Artificial Intelligence Organization, 7 2019.
- [24] Qianli Ma, Sen Li, and Garrison W Cottrell. Adversarial joint-learning recurrent neural network for incomplete time series classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(4):1765–1776, 2020.
- [25] Qianli Ma, Sen Li, Lifeng Shen, Jiabing Wang, Jia Wei, Zhiwen Yu, and Garrison W Cottrell. End-to-end incomplete time-series modeling from linear memory of latent variables. *IEEE transactions on cybernetics*, 50(12):4908–4920, 2019.
- [26] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023.
- [27] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- [28] Zijie Pan, Yushan Jiang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. S2ip-llm: Semantic space informed prompt learning with llm for time series forecasting. ArXiv, abs/2403.05798, 2024.
- [29] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.
- [30] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

- [31] Chenxi Sun, Yaliang Li, Hongyan Li, and linda Qiao. Test: Text prototype aligned embedding to activate llm's ability for time series. *ArXiv*, abs/2308.08241, 2023.
- [32] Xianfeng Tang, Huaxiu Yao, Yiwei Sun, Charu Aggarwal, Prasenjit Mitra, and Suhang Wang. Joint modeling of local and global temporal dynamics for multivariate time series forecasting with missing values. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5956–5963, 2020.
- [33] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. Advances in Neural Information Processing Systems, 34:24804–24816, 2021.
- [34] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In *International Conference on Learning Representations*, 2020.
- [35] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [36] Stef Van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45:1–67, 2011.
- [37] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.
- [38] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The eleventh international conference on learning representations*, 2022.
- [39] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.
- [40] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. *Advances in neural information processing systems*, 29, 2016.
- [41] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.
- [42] Yiyuan Zhang, Xiaohan Ding, Kaixiong Gong, Yixiao Ge, Ying Shan, and Xiangyu Yue. Multimodal pathway: Improve transformers with irrelevant data from other modalities, 2024.
- [43] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pages 27268–27286. PMLR, 2022.
- [44] Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36, 2024.
- [45] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2021.

# **A** Experimental Details

### A.1 Dataset Details

Our research encompass experimental evaluations utilizing a selection of ten popular multivariate datasets, including (1) **ETT**<sup>5</sup> reports seven factors of electricity transformers, encompassing four subsets (ETTm1, ETTm2, ETTh1, ETTh2). These are divided into two categories based on temporal granularity, with data recorded at intervals of 15 minutes (m) and 1 hour (h), respectively. (2) **ECL**<sup>6</sup> records the hourly electricity consumption of 321 customers. (3) **Weather**<sup>7</sup> comprises 21 meteorological factors, recorded at ten-minute intervals throughout the year 2020, including variables such as temperature, humidity. (4) **PEMS** encompasses four datasets (03, 04, 07 and 08), each of which collects data on California's public traffic network at a frequency of every five minutes. For the LargeST [19], which consists of 8,600 sensors and spans from 2017 to 2021, we only selected the 2019 data for pre-training NuwaTS in this paper.

For the eleven datasets mentioned above, we split the data from the variable dimension rather than the time dimension, using a 1:1:1 ratio for training, validation, and test datasets. This approach is necessary because some baselines use a channel-dependent method, which requires fixed input dimensions for time series.

For the general fused datasets, we processed each time variable separately. We sliced time variables from all datasets (ETT,Weather, ECL, PEMS) of varying lengths with a stride of 1, creating a combined dataset with 17.6 million time segments of fixed length. In this study, we set it to 96. These segments were then randomly masked and used for model training.

Using the same approach, we sliced LargeST into 100.1 million segments to train NuwaTS and PatchTST. This was done for zero-shot evaluation and fine-tuning for forecasting tasks on ETT, weather and ECL, ensuring no data leakage.

# A.2 Implementation Details

Regarding the training details of our model compared to others, we mainly followed the imputation task configuration including optimizer, learning rate and early stop strategy in the https://github.com/thuml/Time-Series-Library for fair comparisons. We trained three different PLM backbones: GPT2 [30], BERT [7], and LLaMA [35]. All the comparison models, as well as GPT2 and BERT, were trained on NVIDIA RTX 3090-24G GPUs. LLaMA-version (the first six layers) was trained on an NVIDIA A6000-48G GPU.

# A.3 Different parameter-efficient training strategy

We trained NuwaTS using three different PLM backbones. Table 9 presents the detailed training specifics for each experiment and backbone, including fine-tuned network layers, memory usage, and the number of parameters.

Table 9: Different parameter-efficient training strategy on GPT2, BERT and LLaMA2.

Task	InputSize	Backbone(number of layers)	TrainingModule	MemoryOccupation(GB)	Training/TotalParameter(M)
Imputation	(512,96,1)	GPT2(3)	LN, WPE, IN&OUT, FFN	2.7	20.3/66.0
Imputation	(512,96,1)	GPT2(6)	LN, WPE, IN&OUT, FFN	4.6	38.0/90.8
Imputation	(512,96,1)	GPT2(9)	LN, WPE, IN&OUT, FFN	6.4	55.8/115.6
Imputation	(512,96,1)	GPT2(12)	LN, WPE, IN&OUT, FFN	8.3	73.5/140.4
Imputation	(512,96,1)	LLaMA2(6)	LN, IN&OUT	19.5	6.3/1351.7
Imputation	(128,96,1)	LLaMA2(6)	LN, IN&OUT, FFN	20.3	818.0/1351.7
Imputation	(512,96,1)	BERT(6)	LN, WPE, IN&OUT, FFN	3.1	34.1/68.2
Imputation(fine-tuned)	(512,96,1)	GPT2(6)	DTL	2.4	7.7/90.8
Forecasting(w/o-OutputLayer)	(512,96,1)	GPT2(6)	DTL, LN, WPE	3.8	8.5/90.8
Forecasting(w-OutputLayer)	(512,96,1)	GPT2(6)	DTL, LN, WPE, OUT	3.8	8.9/90.8

LN: LayerNorm Layer, WPE: Word Position Encoding, IN&OUT: EmbedLayer&OutputLayer

FFN: Feed-Foward Network, DTL: Domain-Transfer Layer&prefix

<sup>5</sup>https://github.com/zhouhaoyi/ETDataset

 $<sup>^6</sup> https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams 2011 2014$ 

<sup>&</sup>lt;sup>7</sup>https://www.bgc-jena.mpg.de/wetter/

# Training Algorithm and Mathematical formula

# Algorithm

# Algorithm 1 One-for-all Model Training

- 1: Given the relatively complete time series datasets  $\mathcal{D} = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N\}$  from diverse set of domains.
- 2: for  $\mathbf{x}_i \in \mathbb{R}^L \leftarrow \mathcal{D}$  do
- Generate two random mask  $\mathbf{m}_{i,1} \in \mathbb{R}^L$ ,  $\mathbf{m}_{i,2} \in \mathbb{R}^L$
- $\hat{\mathbf{x}}_{i,1}, \hat{\mathbf{x}}_{i,2} \leftarrow \mathbf{x}_i \times \mathbf{m}_{i,1}, \mathbf{x}_i \times \mathbf{m}_{i,2}$
- $\mathbf{Z}_{i,(p)} \leftarrow \mathrm{Embed}(\hat{\mathbf{x}}_i)$   $\mathbf{E}_{i,(p)} \leftarrow \mathbf{Z}_{i,(p)} + \mathbf{Z}_{i,(v_p)} + \mathbf{z}_{i,(m)} \times \mathbf{r}_i$   $\mathbf{E}_i \leftarrow [\mathbf{k}, \mathbf{z}_{i,(v_g)}, \mathbf{E}_{i,(p)}]$
- Obtain the last hidden states  $\mathbf{h}_i^{(\text{Layer})} \leftarrow \text{PLM}(\mathbf{E}_i)$
- Obatin the imputed series  $\mathbf{o}_i \leftarrow \text{OutputLayer}\left(\mathbf{h}_i^{(\text{Layer})}\right)$ 9:
- Update  $\Phi$  by gradients for  $\mathcal{L}_{mse}\left(\mathbf{o}_{i,1},\mathbf{x}_{i}\right) + \mathcal{L}_{mse}\left(\mathbf{o}_{i,2},\mathbf{x}_{i}\right) + \alpha \mathcal{L}_{cl}\left(\mathbf{h}_{i,1}^{(\text{Layer})},\mathbf{h}_{i,2}^{(\text{Layer})}\right)$ 10:
- 11: **end for**

# Algorithm 2 Domain Specific Fine-tuning

- 1: Initialize continuous prefix  $\mathcal{P} \in \mathbb{R}^{2 \times \text{Layer} \times d}$
- 2: Obtain  $\hat{\mathcal{K}} \in \mathbb{R}^{2 \times \text{Layer} \times d} \leftarrow \text{DomainTrans}(\mathbf{k})$
- 3: WITH NO GRAD:
- 4: **for**  $n \leftarrow 0$  to Layer **do**
- Obtain hidden state  $\mathbf{h}^{(n-1)}$  from PLMLayer<sup>(n-1)</sup>
- Obtain  $[\mathbf{Key}_p, \mathbf{Value}_p] \leftarrow \mathcal{P}^{(n)} + \beta \hat{\mathcal{K}}^{(n)}$  where  $\mathcal{P}^{(n)} \in \mathbb{R}^{2 \times d}$ 6:
- Obtain  $\mathbf{Key} \leftarrow \mathsf{Concat}\left(\mathbf{Key}_p; \mathbf{Key}^{(n)}\right)$
- Obtain  $Value \leftarrow Concat \left(Value_p; Value^{(n)}\right)$ 8:
- Obtain  $\mathbf{h}^{(n)} \leftarrow \text{PLMLayer}^{(n)}(\mathbf{h}^{(n-1)}, \mathbf{Key}, \mathbf{Value})$
- 10: **end for**
- 11: Obtain  $\mathbf{h}^{(Layer)}$

### **Instance Norm**

We applied reversible instance normalization (RevIN) [14] to individual series before splitting them into patches. Each series was normalized to have zero mean and unit standard deviation, with the original mean and standard deviation stored for later reversal, thereby reducing the domain distribution shift caused by non-stationary information. The details are as follows:

$$\mathbb{E}_{t} \left[ \mathbf{x}_{i} \right] = \frac{1}{L} \sum_{j=1}^{L} \mathbf{x}_{i,j},$$

$$\operatorname{Var} \left[ \mathbf{x}_{i} \right] = \frac{1}{L} \sum_{j=1}^{L} \left( \mathbf{x}_{i,j} - \mathbb{E}_{t} \left[ \mathbf{x}_{i} \right] \right)^{2},$$

$$\hat{\mathbf{x}}_{i} = \left( \frac{\mathbf{x}_{i} - \mathbb{E}_{t} \left[ \mathbf{x}_{i} \right]}{\sqrt{\operatorname{Var} \left[ \mathbf{x}_{i} \right] + \epsilon}} \right).$$
(2)

# **Contrastive Learning Loss**

Given a time patch representation as a query q, and a set of keys  $\mathbb{K} = \{k_0, k_1, \ldots\}$ , including a positive target  $k_{+}$ . We employ the InfoNCE [27] as a auxiliary loss function to optimize the model's performance:

$$\mathcal{L}_{cl} = -\log \frac{\exp(q^T W k_+)}{\exp(q^T W k_+) + \sum_{i=0}^{K-1} \exp(q^T W k_i)},$$
(3)

where we employ bi-linear inner-product and  $\boldsymbol{W}$  is a learnable matrix.

# **C** Visualization

# **C.1** Imputation Visualization

We visualized the imputation results of NuwaTS in Figure 6, Figure 7, Figure 8 and Figure 9.

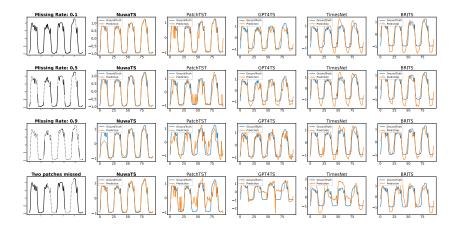


Figure 6: Case visualization when missing rate set to 0.1, 0.5, 0.9. We also visualized the case where the two patches are missing.

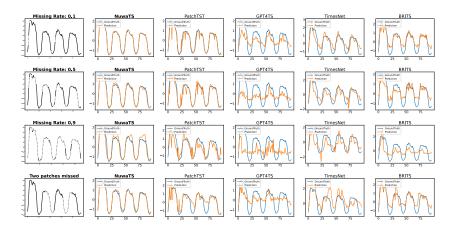


Figure 7: Case visualization when missing rate set to 0.1, 0.5, 0.9. We also visualized the case where the two patches are missing.

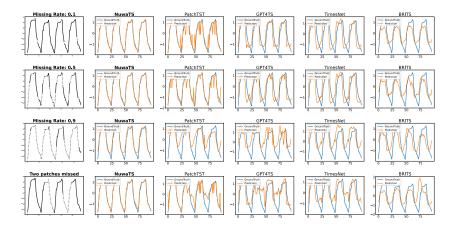


Figure 8: Case visualization when missing rate set to 0.1, 0.5, 0.9. We also visualized the case where the two patches are missing.

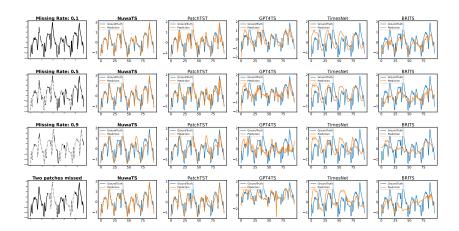


Figure 9: Case visualization when missing rate set to 0.1, 0.5, 0.9. We also visualized the case where the two patches are missing.

# **C.2** Forecasting Visualization

We visualized the forecasting results of fine-tuned NuwaTS whose backbone had not undergone any forecasting training in Figure 10.

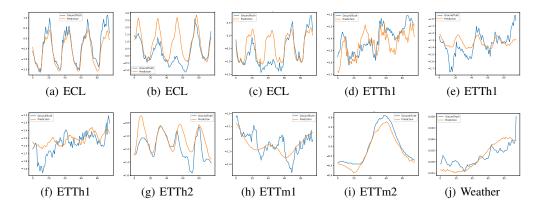


Figure 10: Case visualization when sequence length set to 96 and forecasting length set to 96.

# C.3 Domain Recognition

We also visualized the domain recognition ability of NuwaTS in Figure 11. By directly appending domain-specific embeddings  $\mathbf{k}$  again to the end of the input embedding  $\mathbf{E}_i \in \mathbb{R}^{(N+2)\times D}$ , we get  $\hat{\mathbf{E}}_i = [\mathbf{k}, \mathbf{z}_{i,(v_g)}, \mathbf{E}_{i,(p)}, \mathbf{k}]$ , where  $\hat{\mathbf{E}}_i \in \mathbb{R}^{D\times (N+3)}$ . We feed the  $\hat{\mathbf{E}}_i$  into the NuwaTS (one-for-all) model, and get the final representation  $\hat{\mathbf{h}}_i^{(\text{Layer})} \in \mathbb{R}^{(N+3)\times D}$ . Then We extract the last embeddings  $\mathbf{k}' \in \mathbb{R}^D$  from  $\hat{\mathbf{h}}_i^{(\text{Layer})}$  which is mixed with the patch embeddings in front of the input embeddings, thus exhibiting distinct domain characteristics. We collected N time seires from different domain and extracted  $\mathbf{k}'$ . After applying the t-SNE [37] method to reduce the dimension to 2, we obtained scattered points  $\mathbf{k}'^P \in \mathbb{R}^{N\times 2}$  and visualized them. The scattered points corresponding to time series from the same domain tend to cluster together which indicates that NuwaTS could recognize domain information.

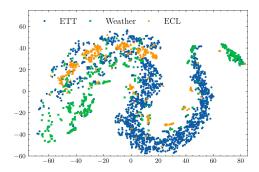


Figure 11: Illustration of domain recognition.

# **D** Extra Experiments

### D.1 Learning From Incomplete Training data

Table 10: When model training on incomplete data with 0.2 and 0.5 original missing rate. The comparison between NuwaTS and PatchTST. A lower MSE or MAE indicates a better imputation performance.

Methods	Ш	ETTh	1	ETT	Th2	ETI	m1	ETT	m2	EC	L	Wea	ther
Methods	I	MSE I	MAE	MSE	MAE								
Default NuwaTS-Origin Missing Rate 0.2 NuwaTS-Origin Missing Rate 0.5		0.169 (	0.263 0.276 0.274	0.018 0.018 0.020	0.084 0.085 0.086	0.064 0.070 0.077	0.147 0.160 0.167	0.010 0.010 0.011	0.060 0.061 0.062	0.085 0.130 0.160	0.186 0.233 0.258	0.206 0.214 0.228	0.088 0.099 0.114
PatchTST PatchTST-Origin Missing Rate 0.2 PatchTST-Origin Missing Rate 0.5		0.185 (	0.288 0.295 0.316	0.019 0.019 0.021	0.088 0.089 0.093	0.075 0.079 0.079	0.172 0.179 0.181	0.011 0.011 0.011	0.065 0.066 0.067	0.121 0.150 0.293	0.243 0.264 0.387	0.230 0.239 0.244	0.116 0.118 0.124

We simulated the training of the model on incomplete time series by randomly omitting 20% and 50% of the original training data and then tested its ability of resisting disturbance. The experimental results indicate that NuwaTS has strong resilience and can achieve good generalization performance when trained on data with a high missing rate (details shown in Table 10).

# D.2 The effect of the number of the GPT2 layer.

Through experiments, we tested the impact of the number of GPT-2 layers and patch size on the final results in Figure 12. The experimental results showed that using 6 GPT-2 layers yielded the best performance.

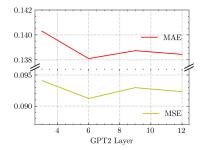


Figure 12: The effect of the number of GPT2 layers on Imputation Performance.

# D.3 Effect of the Type of PLM Backbone

Table 11: Effect of different PLM backbone. A lower MSE or MAE indicates a better imputation performance. **Red**: the best.

Model	ET	Th1		Th2	ET.		ET.			CL	Weather		
Model	MSE	MAE	MSE	MAE									
GPT2	0.164	0.263	0.018	0.084	0.064	0.147	0.010	0.060	0.085	0.186	0.206	0.088	
BERT	0.169	0.267	0.018	0.084	0.065	0.148	0.010	0.060	0.091	0.193	0.217	0.091	
LLaMA2	0.170	0.272	0.018	0.084	0.067	0.152	0.010	0.060	0.096	0.200	0.210	0.090	

We experimented with three different backbones in Table 11: GPT2 [30], BERT [7], and LLaMA [35], uniformly using the first six layers of each. Among them, BERT is a transformer with bi-directional attention, while the other two, GPT2 and LLaMA, employ causal attention.

Using GPT-2 as the backbone outperformed the much larger LLaMA2, likely due to the relatively smaller amount of training data compared to LLaMA2's massive parameter count.

Additionally, LLaMA2's large embedding dimensions produced sparse features and the slow speed of LLaMA2 during training and evaluation makes deployment on edge devices challenging.

BERT, with the similar size as GPT-2, mainly differs in its bidirectional attention. It slightly lags behind GPT-2 due to the causal nature of time series data. However, BERT's bidirectional attention allows the domain-specific embedding to capture domain information from the following embed patches.

Based on the BERT model pre-trained on general fused dataset, We have  $\mathbf{E}_i = [\mathbf{k}, \mathbf{z}_{i,(v_g)}, \mathbf{E}_{i,(p)}]$  before feeding into the backbone, where  $\mathbf{E}_i \in \mathbb{R}^{D \times (N+2)}$ . We feed the  $\mathbf{E}_i$  into the backbone, and get the final representation  $\mathbf{h}_i^{(\text{Layer})} \in \mathbb{R}^{(N+2) \times D}$ . Then We extract the first embedding  $\mathbf{k}' \in \mathbb{R}^D$  from  $\mathbf{h}_i^{(\text{Layer})}$  and take it as [CLS] token [7] which is mixed with the patch embeddings through bidirectional attention mechanism. With same approach as Appendix C.3, we obtain scattered points  $\mathbf{k}'^p \in \mathbb{R}^{N \times 2}$  and visualize them in Figure 13.

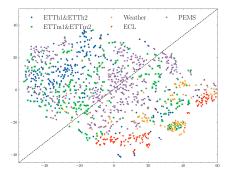


Figure 13: Domain information visualization. We take the model's first output embedding as the [CLS] token, which carries domain information from following sequences. We then use the t-SNE [37] method to visualize the domain information from different datasets.

# D.4 Embed with simple linear layer VS. Embed with text-alignment.

For tokenization, we employ a simple linear layer to embed patches. We acknowledge that recent research [12] has proposed utilizing Patch Reprogramming mechanism to align time patches with PLM's pre-trained word embeddings, thereby activating the model's time series understanding and reasoning capabilities. However, we contend that, due to the high proportion of missing values in the input masked patches and the varying locations of these missing values, modality alignment is not effective in representing such complex incomplete time series. Table 12 shows that simple linear embedding is better than the text-alignment strategy.

Table 12: Embedding methods comparision.

Model	ET	Th1	ET	Th2	ET	Imi	ET	Γm2		CL	Wea	ather
Model	MSE	MAE										
Simple linear layer	0.164	0.263	0.018	0.084	0.064	0.147	0.010	0.060	0.085	0.186	0.206	0.088
Embed with text-alignment [12]	0.250	0.357	0.024	0.100	0.128	0.242	0.014	0.076	0.205	0.311	0.323	0.153

# **E** Main Results in Detailed

We reported detailed results for various baselines and different variants of our model across ten datasets, with missing rates ranging from 0.1 in Table 13 to 0.9 in Table 21.

Table 13: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.1. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	MSE ET	Th1 MAE	MSE ET	Th2 MAE	MSE ETT	fm1 MAE	MSE ET	Γm2 MAE	MSE E	CL MAE	Wea MSE	ther MAE	PEN MSE	IS03 MAE	PEN MSE	IS04 MAE	PEN MSE	IS07 MAE	PEN MSE	IS08 MAE
Median	0.723	0.609	0.725	0.472	0.698	0.582	0.746	0.464	0.992	0.825	0.991	0.494	0.667	0.597	0.716	0.633	0.728	0.634	0.708	0.641
Last	0.399	0.460	0.068	0.135	0.310	0.399	0.039	0.109	0.912	0.811	0.692	0.399	0.448	0.489	0.460	0.501	0.480	0.500	0.420	0.479
Autoformer(2021)	0.487	0.529	0.629	0.509	0.518	0.477	0.231	0.319	0.074	0.194	0.435	0.370	2.698	1.348	0.532	0.583	0.677	0.670	3.714	1.575
Fedformer(2022)	0.276	0.393	1.278	0.645	0.030	0.120	0.023	0.106	0.072	0.193	0.142	0.126	0.151	0.281	0.455	0.503	0.361	0.463	0.252	0.334
Dlinear(2023)	0.284	0.383	0.235	0.330	0.262	0.383	0.399	0.472	0.259	0.403	0.355	0.353	0.204	0.377	0.219	0.388	0.283	0.445	0.315	0.468
iTransformer(2024)	0.447	0.490	0.056	0.147	0.129	0.254	0.074	0.196	0.053	0.153	0.308	0.243	0.075	0.192	0.084	0.204	0.062	0.175	0.097	0.220
BRITS(2018)	0.119	0.226	0.063	0.127	0.046	0.122	0.032	0.084	0.233	0.367	0.798	0.476	0.140	0.263	0.252	0.362	0.227	0.353	0.210	0.331
TimesNet(2022)	0.109	0.230	0.014	0.080	0.034	0.118	0.008	0.056	0.319	0.422	0.617	0.271	0.086	0.200	0.139	0.253	0.119	0.236	0.100	0.213
PatchTST(2023)	0.123	0.246	0.017	0.086	0.059	0.167	0.009	0.061	0.078	0.216	0.193	0.105	0.051	0.166	0.070	0.192	0.044	0.151	0.064	0.184
SAITS(2023)	0.124	0.231	0.188	0.176	0.064	0.130	0.135	0.139	0.392	0.480	0.930	0.492	0.154	0.282	0.264	0.374	0.223	0.346	0.245	0.359
GPT4TS(2024)	0.095	0.207	0.017	0.079	0.028	0.103	0.008	0.053	0.223	0.348	0.968	0.317	0.087	0.205	0.143	0.256	0.116	0.234	0.096	0.215
NuwaTS(specific)	0.123	0.244	0.017	0.086	0.052	0.152	0.009	0.062	0.047	0.147	0.237	0.112	0.040	0.138	0.050	0.148	0.033	0.119	0.054	0.156
PatchTST(one-for-all)	0.116	0.237	0.014	0.080	0.044	0.138	0.009	0.060	0.072	0.199	0.175	0.100	0.047	0.156	0.057	0.169	0.041	0.146	0.051	0.158
NuwaTS(one-for-all)	0.101	0.209	0.014	0.075	0.034	0.113	0.008	0.055	0.046	0.142	0.141	0.075	0.040	0.138	0.052	0.152	0.033	0.121	0.046	0.139
NuwaTS(fine-tuned)	0.100	0.208	0.014	0.076	0.035	0.114	0.008	0.054	0.045	0.141	0.138	0.072	0.041	0.139	0.053	0.155	0.034	0.123	0.046	0.140

Table 14: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.2. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	MSE ET	Th1 MAE	MSE ET	Th2 MAE	MSE ETT	ml MAE	MSE ET	Γm2 MAE	MSE E	CL MAE	Wea MSE	ither MAE	PEN MSE	IS03 MAE	PEN MSE	IS04 MAE	PEN MSE	IS07 MAE	PEN MSE	IS08 MAE
Median	0.713	0.606	0.725	0.471	0.692	0.580	0.741	0.462	0.994	0.825	0.997	0.492	0.679	0.603	0.727	0.638	0.740	0.639	0.719	0.646
Last	0.402	0.461	0.071	0.136	0.312	0.400	0.041	0.110	0.917	0.813	0.703	0.340	0.450	0.490	0.463	0.502	0.482	0.502	0.422	0.480
Autoformer(2021)	0.425	0.493	0.532	0.472	0.345	0.407	0.139	0.263	0.082	0.205	0.423	0.342	1.168	0.880	0.218	0.356	0.250	0.384	1.725	1.061
Fedformer(2022)	0.270	0.388	0.806	0.536	0.043	0.145	0.026	0.113	0.080	0.204	0.162	0.138	0.151	0.285	0.357	0.445	0.293	0.410	0.223	0.323
Dlinear(2023)	0.215	0.331	0.120	0.227	0.153	0.284	0.219	0.345	0.179	0.323	0.263	0.253	0.114	0.266	0.126	0.280	0.156	0.319	0.178	0.341
iTransformer(2024)	0.480	0.507	0.101	0.199	0.154	0.279	0.104	0.235	0.058	0.160	0.344	0.277	0.080	0.199	0.091	0.216	0.068	0.186	0.107	0.235
BRITS(2018) TimesNet(2022) PatchTST(2023) SAITS(2023) GPT4TS(2024) NuwaTS(specific)	0.131	0.241	0.069	0.133	0.050	0.128	0.036	0.089	0.243	0.375	0.779	0.474	0.140	0.263	0.252	0.363	0.226	0.352	0.212	0.332
	0.114	0.235	0.015	0.081	0.036	0.120	0.008	0.057	0.323	0.425	0.635	0.272	0.088	0.203	0.140	0.255	0.120	0.238	0.102	0.215
	0.131	0.253	0.018	0.087	0.059	0.164	0.009	0.061	0.082	0.219	0.198	0.107	0.050	0.162	0.068	0.187	0.043	0.150	0.062	0.177
	0.131	0.238	0.177	0.169	0.063	0.131	0.132	0.125	0.405	0.488	0.935	0.492	0.154	0.283	0.264	0.375	0.224	0.346	0.245	0.360
	0.111	0.221	0.018	0.081	0.032	0.110	0.009	0.055	0.233	0.356	0.973	0.317	0.088	0.206	0.144	0.258	0.118	0.236	0.096	0.213
	0.129	0.250	0.017	0.086	0.049	0.144	0.009	0.060	0.049	0.149	0.241	0.114	0.040	0.137	0.049	0.147	0.032	0.118	0.051	0.150
PatchTST(one-for-all) NuwaTS(one-for-all) NuwaTS(fine-tuned)	0.120 0.103 0.101	0.240 0.209 0.207	0.015 0.014 0.014	0.080 0.075 0.075	0.045 0.035	0.139 0.111 0.111	0.009 0.008 0.008	0.060 0.054 0.053	0.075 0.049 0.047	0.202 0.145 0.144	0.177 0.148 0.140	0.100 0.072 0.071	0.047 0.038 0.039	0.157 0.135 0.135	0.057 0.049 0.050	0.169 0.148 0.149	0.041 0.032 0.032	0.146 0.118 0.118	0.051 0.044 0.044	0.157 0.136 0.136

Table 15: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.3. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	ET		ET		ETT		ET		E			ither	PEM		PEM		PEN			1S08
	MSE	MAE																		
Median	0.707	0.605	0.717	0.469	0.689	0.579	0.740	0.462	0.988	0.826	0.991	0.492	0.681	0.604	0.730	0.639	0.742	0.640	0.722	0.647
Last	0.406	0.463	0.072	0.138	0.314	0.401	0.043	0.111	0.922	0.814	0.703	0.341	0.452	0.492	0.465	0.504	0.485	0.504	0.425	0.482
Autoformer(2021)	0.409	0.481	0.423	0.418	0.279	0.373	0.120	0.248	0.091	0.217	0.444	0.359	0.664	0.644	0.190	0.336	0.188	0.334	1.012	0.790
Fedformer(2022)	0.271	0.388	0.574	0.468	0.038	0.136	0.029	0.121	0.088	0.216	0.162	0.149	0.157	0.294	0.296	0.405	0.248	0.374	0.206	0.319
Dlinear(2023)	0.187	0.307	0.061	0.159	0.098	0.221	0.112	0.243	0.143	0.279	0.200	0.181	0.074	0.202	0.083	0.214	0.088	0.227	0.103	0.246
iTransformer(2024)	0.517	0.523	0.155	0.250	0.186	0.307	0.147	0.281	0.062	0.167	0.366	0.313	0.086	0.209	0.100	0.227	0.075	0.196	0.120	0.250
BRITS(2018)	0.143	0.254	0.076	0.138	0.055	0.135	0.040	0.094	0.254	0.384	0.783	0.474	0.140	0.263	0.253	0.363	0.225	0.351	0.214	0.334
TimesNet(2022)	0.120	0.241	0.017	0.083	0.038	0.123	0.009	0.057	0.329	0.428	0.652	0.273	0.090	0.206	0.142	0.257	0.121	0.239	0.103	0.217
PatchTST(2023)	0.138	0.259	0.018	0.088	0.059	0.163	0.009	0.062	0.086	0.222	0.213	0.108	0.050	0.160	0.066	0.183	0.043	0.149	0.060	0.173
SAITS(2023)	0.139	0.246	0.180	0.169	0.064	0.135	0.132	0.124	0.417	0.495	0.947	0.490	0.155	0.283	0.266	0.376	0.224	0.347	0.246	0.360
GPT4TS(2024)	0.124	0.235	0.019	0.083	0.038	0.118	0.009	0.056	0.244	0.364	0.953	0.317	0.089	0.207	0.146	0.260	0.119	0.238	0.097	0.212
NuwaTS(specific)	0.136	0.257	0.017	0.086	0.048	0.140	0.009	0.059	0.052	0.153	0.251	0.115	0.040	0.138	0.049	0.148	0.032	0.118	0.049	0.146
PatchTST(one-for-all)	0.126	0.245	0.015	0.080	0.047	0.142	0.009	0.060	0.079	0.206	0.174	0.101	0.047	0.157	0.057	0.170	0.042	0.147	0.052	0.158
NuwaTS(one-for-all)	0.109	0.214	0.014	0.076	0.036	0.113	0.008	0.054	0.052	0.150	0.152	0.073	0.038	0.134	0.049	0.148	0.032	0.118	0.044	0.135
NuwaTS(fine-tuned)	0.106	0.211	0.014	0.075	0.036	0.113	0.008	0.053	0.050	0.148	0.146	0.071	0.038	0.134	0.049	0.148	0.032	0.118	0.044	0.135

Table 16: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.4. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	MSE ET	Th1 MAE	MSE ET	Th2 MAE	MSE ETT	fm1 MAE	MSE ET	Γm2 MAE	MSE E	CL MAE	MSE Wea	ther MAE	PEN MSE	IS03 MAE	PEN MSE	IS04 MAE	PEN MSE	IS07 MAE	PEN MSE	MAE
Median	0.710	0.606	0.716	0.469	0.687	0.579	0.736	0.461	0.988	0.827	0.984	0.493	0.688	0.607	0.736	0.642	0.749	0.643	0.728	0.650
Last	0.411	0.466	0.073	0.139	0.318	0.403	0.045	0.113	0.930	0.816	0.701	0.343	0.455	0.494	0.468	0.506	0.488	0.506	0.428	0.484
Autoformer(2021)	0.426	0.487	0.347	0.367	0.238	0.349	0.113	0.239	0.103	0.232	0.501	0.393	0.419	0.500	0.231	0.374	0.213	0.356	0.632	0.608
Fedformer(2022)	0.280	0.395	0.447	0.421	0.053	0.164	0.033	0.129	0.100	0.231	0.174	0.163	0.171	0.310	0.255	0.378	0.217	0.350	0.199	0.321
Dlinear(2023)	0.194	0.311	0.046	0.138	0.084	0.200	0.055	0.167	0.144	0.275	0.182	0.149	0.073	0.200	0.080	0.207	0.064	0.185	0.073	0.195
iTransformer(2024)	0.565	0.542	0.230	0.309	0.235	0.345	0.213	0.341	0.069	0.175	0.412	0.357	0.094	0.221	0.110	0.240	0.083	0.208	0.135	0.266
BRITS(2018)	0.162	0.273	0.086	0.147	0.062	0.145	0.047	0.101	0.270	0.395	0.781	0.472	0.140	0.264	0.253	0.364	0.224	0.350	0.217	0.336
TimesNet(2022)	0.128	0.249	0.017	0.085	0.041	0.128	0.009	0.059	0.336	0.433	0.664	0.274	0.093	0.210	0.144	0.259	0.123	0.242	0.106	0.221
PatchTST(2023)	0.148	0.269	0.019	0.089	0.061	0.163	0.009	0.062	0.093	0.227	0.218	0.11	0.050	0.160	0.064	0.180	0.044	0.148	0.058	0.169
SAITS(2023)	0.150	0.257	0.191	0.174	0.066	0.139	0.132	0.126	0.429	0.503	0.936	0.486	0.155	0.284	0.267	0.377	0.226	0.347	0.246	0.361
GPT4TS(2024)	0.143	0.252	0.021	0.085	0.045	0.128	0.010	0.058	0.258	0.375	0.945	0.317	0.090	0.208	0.149	0.262	0.122	0.240	0.099	0.214
NuwaTS(specific)	0.145	0.265	0.017	0.086	0.050	0.140	0.009	0.059	0.058	0.160	0.264	0.117	0.041	0.139	0.050	0.149	0.033	0.119	0.049	0.145
PatchTST(one-for-all) NuwaTS(one-for-all) NuwaTS(fine-tuned)	0.136 0.119	0.254 0.225 0.220	0.016 0.015	0.082 0.077 0.076	0.051 0.040 0.040	0.147 0.118 0.117	0.009 0.009	0.060 0.055 0.054	0.085 0.057	0.212 0.157 0.155	0.187 0.162 0.168	0.104 0.075 0.074	0.048 0.039	0.159 0.135 0.135	0.058 0.050 0.050	0.171 0.148 0.148	0.043 0.033	0.148 0.119 0.119	0.052 0.044	0.159 0.136 0.136

Table 17: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.5. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	MSE ET	Th1 MAE	MSE ET	Th2 MAE	MSE ETT	ſm1 MAE	MSE ET	Γm2 MAE	MSE E	CL MAE	Wea MSE	ther MAE	PEN MSE	IS03 MAE	PEN MSE	IS04 MAE	PEN MSE	IS07 MAE	PEM MSE	IS08 MAE
Median	0.708	0.605	0.721	0.470	0.686	0.578	0.735	0.461	0.994	0.830	0.991	0.493	0.669	0.597	0.721	0.634	0.731	0.634	0.712	0.642
Last	0.416	0.469	0.078	0.142	0.323	0.406	0.049	0.115	0.941	0.820	0.716	0.345	0.460	0.498	0.474	0.509	0.493	0.509	0.433	0.487
Autoformer(2021)	0.469	0.507	0.320	0.336	0.223	0.339	0.116	0.237	0.119	0.249	0.571	0.432	0.329	0.441	0.301	0.432	0.276	0.408	0.461	0.515
Fedformer(2022)	0.298	0.407	0.389	0.394	0.061	0.176	0.038	0.139	0.114	0.248	0.192	0.182	0.197	0.336	0.241	0.371	0.210	0.346	0.203	0.331
Dlinear(2023)	0.239	0.345	0.084	0.183	0.122	0.240	0.062	0.177	0.189	0.319	0.212	0.186	0.120	0.268	0.125	0.272	0.094	0.233	0.099	0.234
iTransformer(2024)	0.620	0.564	0.321	0.372	0.297	0.390	0.299	0.405	0.077	0.187	0.473	0.405	0.104	0.234	0.122	0.253	0.093	0.220	0.151	0.283
BRITS(2018) TimesNet(2022) PatchTST(2023) SAITS(2023) GPT4TS(2024) NuwaTS(specific)	0.184	0.294	0.097	0.157	0.072	0.158	0.055	0.109	0.290	0.410	0.790	0.471	0.141	0.264	0.255	0.366	0.223	0.349	0.221	0.339
	0.140	0.260	0.019	0.087	0.046	0.135	0.010	0.060	0.345	0.439	0.683	0.276	0.096	0.214	0.146	0.262	0.126	0.245	0.109	0.225
	0.162	0.281	0.020	0.090	0.065	0.166	0.010	0.063	0.102	0.236	0.239	0.113	0.051	0.161	0.064	0.178	0.045	0.149	0.059	0.168
	0.166	0.271	0.203	0.180	0.070	0.146	0.134	0.131	0.441	0.510	0.935	0.483	0.156	0.284	0.269	0.379	0.227	0.348	0.248	0.362
	0.169	0.273	0.022	0.088	0.056	0.141	0.011	0.060	0.275	0.387	0.939	0.317	0.093	0.211	0.152	0.265	0.124	0.243	0.102	0.217
	0.158	0.277	0.018	0.087	0.054	0.144	0.009	0.060	0.065	0.171	0.280	0.120	0.043	0.141	0.052	0.151	0.035	0.122	0.049	0.146
PatchTST(one-for-all)	0.150	0.267	0.017	0.084	0.057	0.155	0.010	0.062	0.093	0.221	0.195	0.107	0.050	0.161	0.060	0.172	0.044	0.150	0.054	0.161
NuwaTS(one-for-all)	0.135	0.241	0.016	0.080	0.045	0.126	0.009	0.056	0.064	0.167	0.180	0.079	0.041	0.138	0.051	0.150	0.034	0.121	0.046	0.138
NuwaTS(fine-tuned)	0.128	0.233	0.015	0.078	0.045	0.124	0.009	0.055	0.062	0.164	0.188	0.077	0.041	0.138	0.051	0.150	0.034	0.121	0.046	0.138

Table 18: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.6. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	ET			Th2	ET			Γm2	E			ither		1S03	PEM		PEN		PEM	
Woder	MSE	MAE																		
Median	0.710	0.607	0.718	0.470	0.689	0.580	0.736	0.461	1.005	0.829	0.990	0.495	0.686	0.606	0.735	0.642	0.747	0.642	0.728	0.650
Last	0.425	0.473	0.083	0.145	0.329	0.409	0.054	0.119	0.955	0.824	0.725	0.347	0.467	0.502	0.480	0.513	0.499	0.513	0.439	0.491
Autoformer(2021)	0.526	0.535	0.335	0.335	0.234	0.347	0.136	0.254	0.137	0.269	0.640	0.469	0.322	0.439	0.378	0.489	0.349	0.463	0.408	0.487
Fedformer(2022)	0.323	0.423	0.348	0.370	0.060	0.172	0.045	0.151	0.131	0.266	0.216	0.204	0.235	0.369	0.254	0.383	0.230	0.362	0.218	0.346
Dlinear(2023)	0.313	0.397	0.163	0.261	0.198	0.313	0.125	0.254	0.265	0.390	0.279	0.262	0.203	0.364	0.206	0.365	0.169	0.327	0.170	0.324
iTransformer(2024)	0.674	0.585	0.418	0.430	0.367	0.437	0.394	0.467	0.087	0.198	0.539	0.450	0.113	0.245	0.135	0.267	0.103	0.232	0.166	0.297
BRITS(2018)	0.210	0.318	0.112	0.169	0.085	0.175	0.066	0.119	0.314	0.427	0.791	0.471	0.141	0.265	0.257	0.369	0.224	0.348	0.225	0.343
TimesNet(2022)	0.155	0.273	0.020	0.090	0.053	0.144	0.010	0.062	0.356	0.446	0.696	0.277	0.100	0.219	0.149	0.266	0.129	0.249	0.112	0.230
PatchTST(2023)	0.180	0.297	0.021	0.092	0.071	0.173	0.010	0.065	0.115	0.249	0.259	0.118	0.053	0.164	0.066	0.180	0.047	0.152	0.060	0.170
SAITS(2023)	0.185	0.287	0.217	0.188	0.076	0.156	0.135	0.138	0.453	0.517	0.934	0.480	0.156	0.285	0.271	0.380	0.228	0.349	0.249	0.364
GPT4TS(2024)	0.197	0.295	0.025	0.092	0.069	0.157	0.012	0.063	0.293	0.401	0.924	0.318	0.096	0.215	0.155	0.269	0.128	0.247	0.106	0.222
NuwaTS(specific)	0.175	0.291	0.019	0.088	0.060	0.151	0.010	0.062	0.075	0.183	0.297	0.123	0.045	0.145	0.054	0.154	0.037	0.125	0.051	0.148
PatchTST(one-for-all)	0.169	0.284	0.018	0.086	0.065	0.165	0.010	0.063	0.104	0.232	0.209	0.111	0.052	0.163	0.062	0.175	0.046	0.152	0.056	0.163
NuwaTS(one-for-all)	0.155	0.260	0.017	0.083	0.053	0.137	0.010	0.058	0.073	0.179	0.200	0.084	0.043	0.141	0.054	0.154	0.036	0.124	0.048	0.142
NuwaTS(fine-tuned)	0.146	0.249	0.016	0.080	0.051	0.134	0.010	0.057	0.070	0.176	0.202	0.082	0.043	0.141	0.053	0.153	0.036	0.124	0.048	0.141

Table 19: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.7. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	ET.	Γh1	ET	Th2	ETT	ſm1	ET	Tm2	E	CL	Wes	ither	PEN	1S03	PEM	1S04	PEN	1S07	PEN	4S08
Model	MSE	MAE																		
Median	0.717	0.610	0.718	0.470	0.694	0.583	0.744	0.463	1.015	0.833	1.002	0.499	0.710	0.623	0.758	0.657	0.775	0.658	0.750	0.665
Last	0.438	0.480	0.093	0.151	0.341	0.416	0.062	0.125	0.980	0.830	0.740	0.351	0.478	0.509	0.491	0.520	0.511	0.520	0.450	0.498
Autoformer(2021)	0.612	0.576	0.402	0.368	0.284	0.382	0.201	0.311	0.165	0.295	0.725	0.514	0.376	0.476	0.477	0.556	0.446	0.531	0.429	0.502
Fedformer(2022)	0.369	0.449	0.288	0.332	0.077	0.196	0.057	0.171	0.155	0.289	0.256	0.238	0.303	0.421	0.305	0.420	0.292	0.407	0.261	0.379
Dlinear(2023)	0.432	0.470	0.301	0.366	0.331	0.417	0.256	0.370	0.391	0.491	0.397	0.363	0.339	0.487	0.341	0.484	0.303	0.454	0.301	0.449
iTransformer(2024)	0.742	0.613	0.544	0.497	0.466	0.499	0.518	0.537	0.102	0.216	0.627	0.502	0.122	0.256	0.152	0.283	0.117	0.247	0.184	0.313
BRITS(2018)	0.251	0.353	0.135	0.189	0.107	0.203	0.082	0.136	0.351	0.452	0.793	0.471	0.143	0.267	0.262	0.373	0.226	0.350	0.232	0.349
TimesNet(2022)	0.180	0.296	0.023	0.094	0.066	0.160	0.011	0.065	0.374	0.458	0.710	0.280	0.105	0.227	0.155	0.273	0.135	0.257	0.118	0.237
PatchTST(2023)	0.208	0.320	0.023	0.095	0.082	0.186	0.011	0.067	0.139	0.273	0.292	0.126	0.058	0.170	0.070	0.185	0.051	0.159	0.065	0.176
SAITS(2023)	0.217	0.312	0.233	0.199	0.089	0.172	0.137	0.146	0.467	0.525	0.925	0.477	0.158	0.286	0.274	0.383	0.230	0.350	0.251	0.366
GPT4TS(2024)	0.239	0.328	0.028	0.097	0.092	0.182	0.013	0.067	0.320	0.420	0.918	0.320	0.102	0.223	0.161	0.275	0.134	0.254	0.112	0.229
NuwaTS(specific)	0.202	0.312	0.021	0.092	0.072	0.166	0.011	0.064	0.091	0.202	0.338	0.130	0.049	0.151	0.058	0.160	0.040	0.131	0.055	0.154
PatchTST(one-for-all)	0.199	0.308	0.020	0.091	0.079	0.182	0.011	0.066	0.122	0.249	0.232	0.118	0.056	0.168	0.066	0.179	0.049	0.157	0.060	0.168
NuwaTS(one-for-all)	0.188	0.289	0.019	0.087	0.067	0.157	0.011	0.061	0.090	0.198	0.229	0.092	0.047	0.147	0.058	0.159	0.040	0.130	0.052	0.147
NuwaTS(fine-tuned)	0.175	0.275	0.018	0.084	0.064	0.150	0.011	0.060	0.086	0.195	0.231	0.089	0.047	0.147	0.057	0.159	0.040	0.130	0.052	0.147

Table 20: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.8. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	ET	Th1	ET	Th2	ETT	Γm1	ET	Γm2	E	CL	Wea	ither	PEN	1S03	PEM	1S04	PEM	IS07	PEM	IS08
Wiodei	MSE	MAE																		
Median	0.734	0.616	0.740	0.475	0.709	0.588	0.749	0.465	1.023	0.844	1.005	0.502	0.708	0.617	0.762	0.653	0.774	0.654	0.753	0.662
Last	0.463	0.491	0.110	0.162	0.362	0.427	0.077	0.134	1.019	0.840	0.766	0.358	0.498	0.521	0.511	0.532	0.531	0.532	0.469	0.510
Autoformer(2021)	0.724	0.624	0.523	0.432	0.381	0.443	0.311	0.398	0.200	0.324	0.811	0.557	0.477	0.540	0.581	0.620	0.551	0.598	0.514	0.552
Fedformer(2022)	0.448	0.491	0.264	0.304	0.123	0.248	0.078	0.198	0.185	0.315	0.327	0.295	0.403	0.490	0.398	0.483	0.399	0.481	0.348	0.441
Dlinear(2023)	0.574	0.544	0.480	0.469	0.499	0.521	0.465	0.506	0.536	0.586	0.557	0.469	0.502	0.605	0.502	0.598	0.470	0.576	0.468	0.571
iTransformer(2024)	0.810	0.642	0.671	0.557	0.581	0.562	0.647	0.602	0.127	0.242	0.720	0.552	0.128	0.262	0.170	0.300	0.132	0.263	0.208	0.334
BRITS(2018)	0.305	0.394	0.170	0.216	0.144	0.244	0.108	0.160	0.400	0.484	0.799	0.473	0.146	0.271	0.269	0.379	0.232	0.355	0.241	0.357
TimesNet(2022)	0.221	0.329	0.028	0.101	0.093	0.191	0.013	0.070	0.401	0.474	0.724	0.286	0.113	0.237	0.162	0.281	0.143	0.267	0.125	0.246
PatchTST(2023)	0.248	0.351	0.027	0.101	0.102	0.208	0.013	0.071	0.186	0.316	0.341	0.137	0.067	0.180	0.080	0.196	0.059	0.170	0.073	0.185
SAITS(2023)	0.269	0.347	0.253	0.214	0.111	0.198	0.139	0.155	0.484	0.534	0.924	0.477	0.160	0.289	0.277	0.385	0.232	0.352	0.255	0.369
GPT4TS(2024)	0.293	0.367	0.032	0.104	0.127	0.218	0.015	0.072	0.358	0.446	0.911	0.325	0.112	0.235	0.169	0.284	0.143	0.265	0.123	0.242
NuwaTS(specific)	0.242	0.343	0.024	0.097	0.092	0.189	0.013	0.068	0.119	0.231	0.372	0.139	0.057	0.160	0.066	0.169	0.047	0.141	0.062	0.164
PatchTST(one-for-all)	0.243	0.342	0.023	0.098	0.104	0.208	0.013	0.072	0.153	0.275	0.284	0.132	0.064	0.176	0.073	0.188	0.056	0.165	0.067	0.176
NuwaTS(one-for-all)	0.236	0.327	0.023	0.094	0.093	0.187	0.012	0.066	0.116	0.226	0.267	0.104	0.054	0.157	0.065	0.169	0.046	0.140	0.059	0.157
NuwaTS(fine-tuned)	0.218	0.310	0.021	0.090	0.085	0.175	0.012	0.064	0.112	0.223	0.269	0.100	0.054	0.156	0.064	0.168	0.046	0.139	0.058	0.156

Table 21: Detailed Imputation results on ETTs, ECL and weather with missing rate set to 0.9. We set the input length to 96. A lower MSE or MAE indicates a better imputation performance.

Model	MSE ET	Th1 MAE	MSE ET	Th2 MAE	MSE	fm1 MAE	MSE ET	Γm2 MAE	MSE E	CL MAE	Wea MSE	ther MAE	PEN MSE	IS03 MAE	PEN MSE	IS04 MAE	PEN MSE	IS07 MAE	PEN MSE	MAE
Median	0.786	0.635	0.768	0.483	0.748	0.603	0.772	0.472	1.086	0.870	1.027	0.513	0.733	0.626	0.795	0.665	0.805	0.667	0.788	0.676
Last	0.526	0.521	0.156	0.191	0.419	0.456	0.121	0.164	1.112	0.863	0.834	0.377	0.555	0.556	0.569	0.566	0.590	0.566	0.526	0.543
Autoformer(2021)	0.889	0.690	0.735	0.544	0.612	0.563	0.535	0.541	0.268	0.372	0.937	0.616	0.676	0.654	0.731	0.705	0.704	0.688	0.718	0.657
Fedformer(2022)	0.650	0.592	0.445	0.393	0.194	0.307	0.168	0.277	0.239	0.355	0.500	0.402	0.596	0.615	0.592	0.607	0.609	0.618	0.557	0.575
Dlinear(2023)	0.768	0.634	0.711	0.579	0.721	0.635	0.699	0.625	0.748	0.707	0.749	0.567	0.728	0.738	0.712	0.720	0.707	0.715	0.706	0.709
iTransformer(2024)	0.895	0.679	0.826	0.625	0.754	0.648	0.810	0.675	0.196	0.304	0.842	0.610	0.168	0.310	0.215	0.344	0.160	0.294	0.292	0.417
BRITS(2018)	0.415	0.464	0.244	0.277	0.245	0.336	0.171	0.218	0.502	0.548	0.825	0.483	0.155	0.282	0.281	0.389	0.246	0.369	0.258	0.371
TimesNet(2022)	0.328	0.407	0.039	0.115	0.191	0.278	0.018	0.081	0.478	0.520	0.750	0.307	0.145	0.269	0.188	0.306	0.170	0.293	0.152	0.274
PatchTST(2023)	0.328	0.404	0.038	0.114	0.159	0.260	0.018	0.082	0.344	0.436	0.482	0.172	0.098	0.211	0.116	0.230	0.094	0.205	0.104	0.216
SAITS(2023)	0.386	0.416	0.293	0.241	0.176	0.263	0.140	0.170	0.516	0.554	0.926	0.482	0.166	0.295	0.283	0.390	0.236	0.356	0.262	0.374
GPT4TS(2024)	0.390	0.431	0.043	0.118	0.213	0.294	0.020	0.083	0.460	0.511	0.925	0.338	0.148	0.271	0.200	0.313	0.176	0.296	0.158	0.278
NuwaTS(specific)	0.329	0.402	0.034	0.110	0.153	0.248	0.017	0.078	0.219	0.320	0.466	0.170	0.085	0.192	0.095	0.201	0.072	0.172	0.090	0.196
PatchTST(one-for-all)	0.344	0.410	0.034	0.113	0.179	0.275	0.019	0.086	0.308	0.389	0.440	0.173	0.093	0.206	0.105	0.218	0.085	0.196	0.094	0.204
NuwaTS(one-for-all)	0.331	0.394	0.032	0.109	0.172	0.264	0.018	0.080	0.219	0.309	0.377	0.141	0.083	0.189	0.095	0.201	0.074	0.174	0.086	0.188
NuwaTS(fine-tuned)	0.312	0.379	0.030	0.105	0.149	0.239	0.016	0.076	0.201	0.301	0.379	0.136	0.080	0.186	0.092	0.198	0.071	0.171	0.083	0.185

# F Limitation

The model was trained on time series segments fixed at a length of 96, imputing the masked series based on the unmasked portions. Although our model can adapt to different domains and missing patterns, showing strong adaptability after domain-transfer fine-tuning, in practical applications, the

model may require further fine-tuning when dealing with segments longer than 96 as well as the segments that entirely missed. Additionally, NuwaTS discards multivariate correlations, relying on temporal information and identified domain information, which may degrade the imputation performance. For instance, if a sensor loses a large amount of data, the data from adjacent sensors can be used as supplementary information. Future work will aim to improve NuwaTS's capability in handling longer missing segments and incorporate multivariate correlations.

# **G** Broader Impact

We believe that our NuwaTS model not only plays an important role in the time series domain but also that its one-for-all training paradigm and domain-specific fine-tuning method will inspire broader applications in the field of data mining. In practical applications, NuwaTS can be widely used to address various time series data gaps, and the imputed data can enhance the performance of downstream tasks. However, there may be some discrepancies between the time series imputed by NuwaTS and the real data. Therefore, it is necessary to validate the improvements in downstream tasks through multiple methods. Additionally, NuwaTS is not recommended for use in investment-related forecasting, such as stock trading.