TiFuse: Selective and Stable Time–Frequency Fusion for Cloud Workload Forecasting

Anonymous authors
Paper under double-blind review

Abstract

Cloud workloads combine recurring periodicities with abrupt shifts and transient disturbances, requiring models to capture global structures and local dynamics simultaneously. Existing time–frequency forecasters typically learn both domains exhaustively and fuse them directly, which can preserve redundant information and make cross-domain interactions less stable. We propose TiFuse, a selective and stable time–frequency framework for cloud workload forecasting. TiFuse constructs a compact spectral representation by retaining energy-dominant DCT components, and a complementary temporal representation by selecting representative multi-scale historical patches while preserving recent dynamics. Their interaction is modeled with relation-enhanced attention that couples sample-specific dependencies with a global relational prior under input-dependent sparse constraints. Experiments on five real-world cloud traces covering CPU, memory, and GPU workloads show that TiFuse achieves the lowest average MSE in 15 of 16 dataset–resource settings and the lowest average MAE in 13 of 16, while maintaining competitive computational efficiency.

1 Introduction

As cloud services, online applications, and artificial intelligence training workloads continue to expand, cloud data centers must dynamically balance resource utilization, service quality, and operating costs. Continual changes in user access patterns, job submissions, resource contention, and scheduling policies make cloud workload data highly complex (Gao et al., 2025) (Zhao et al., 2025) (Lu et al., 2025). Cloud workloads exhibit recurring periodic patterns, abrupt changes, and transient perturbations. A single-domain representation therefore struggles to capture both their global structure and local dynamics (Huang et al., 2025) (Liu et al., 2025). These characteristics motivate the joint use of time-domain and frequency-domain information to construct complementary dual-domain representations for cloud workloads. However, several challenges remain.

Constructing compact frequency representations remains challenging. Frequency-domain modeling has increasingly been used to learn time-series representations in recent years. The discrete Fourier transform (DFT) is a widely used frequency-domain transformation tool that typically produces complex-valued coefficients. Using these coefficients in real-valued neural networks requires explicit choices about how to represent and process real and imaginary components, or magnitude and phase. Architectures that return to the time domain after frequency-domain transformations also require the corresponding inverse transform (Kou et al., 2025) (Huang et al., 2025). These requirements complicate neural network design and introduce additional computational overhead. Moreover, spectral leakage in the DFT and Gibbs oscillations during truncated reconstruction (Li et al., 2026) (Jiang et al., 2023) hinder the efficient construction of compact frequency-domain representations. As shown in Figure 1, the DCT offers better energy compaction than the DFT (Tew et al., 2026). It can generally approximate the original signal more accurately using fewer low-frequency coefficients while reducing boundary distortion and oscillations during truncated reconstruction.

Extracting historical information from the time domain that complements the frequency domain is also challenging. Jointly using time-domain and frequency-domain representations can alleviate the limitations of either domain alone. Recent studies have combined information from both domains through dual-branch architectures and feature concatenation (Sun et al., 2025). Others have used temporal patch modeling and frequency-domain filtering to capture local correlations and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

global patterns, respectively (Ma et al., 2026). However, more efficient joint use of the two domains requires further investigation. First, real-world time series contain noise and anomalous perturbations. Indiscriminate learning from uncertain or anomalous time steps may limit model adaptability and generalization (Sun et al., 2025; Fu et al., 2025). Second, time-domain and frequency-domain representations are constructed from the same observed sequence. Preventing the two branches from redundantly learning similar patterns therefore remains an important issue in dual-domain collaboration.

Reliably modeling relationships between heterogeneous time-domain and frequency-domain signals remains challenging. Standard dot-product attention does not explicitly adapt to local distributional structure, making it susceptible to local noise and transient perturbations (Kim et al., 2026) (Li et al., 2019). Time-domain and frequency-domain representations differ in their statistical distributions, numerical scales, and semantic emphasis. These differences further complicate stable modeling of within-domain and cross-domain dependencies (Zhang et al., 2022). Consequently, standard dot-product attention struggles to capture both within-domain structure and cross-domain relationships in a unified feature space, reducing the stability of relationship learning.

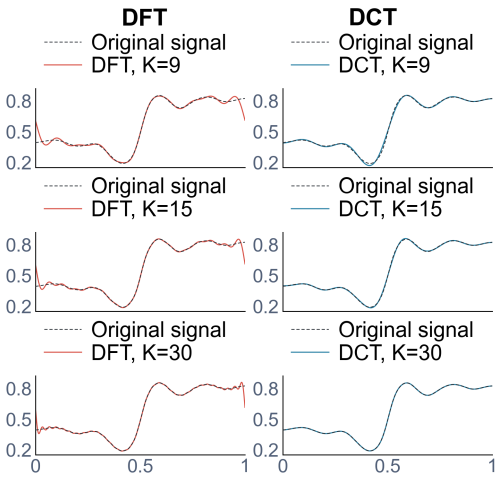


Figure 1: DFT (red) and DCT (blue) reconstructions for $K = 9, 15, 30$ (top to bottom); dashed curves denote the original signal. At $K = 9$, DCT closely follows the signal, whereas DFT exhibits visible boundary deviations.

Based on this analysis, we propose TiFuse, a selective and stable time–frequency framework for cloud workload forecasting. TiFuse uses the DCT to construct compact, real-valued frequency representations. This addresses the architectural complexity, additional computational overhead, and reconstruction accuracy issues associated with complex-valued DFT spectra. To address redundant, anomalous, and repeated information in time-domain inputs, TiFuse selects representative workload patterns from patches at multiple scales. To overcome the limitations of standard dot-product attention in modeling heterogeneous time–frequency dependencies, it further introduces a global relational prior and a sparse matching mechanism. These components improve the stability of time–frequency dependency modeling.

The main contributions of this work are as follows.

1. We propose TiFuse, a selective and stable time–frequency framework that explicitly controls what information each domain contributes and how the two domains interact.
2. We develop a selective dual-domain representation strategy that retains energy-dominant spectral components and representative multi-scale temporal patterns, yielding compact and complementary time–frequency representations.
3. We introduce relation-enhanced attention that combines sample-specific dependencies with a global relational prior under input-dependent sparse constraints, improving the stability of heterogeneous time–frequency interactions.
4. Extensive experiments on five real-world cloud traces covering CPU, memory, and GPU workloads show that TiFuse achieves the lowest average MSE in 15 of 16 dataset–resource settings and the lowest average MAE in 13 of 16, while maintaining competitive computational efficiency.

2 Related Work

2.1 Frequency Representation for Time Series Forecasting

Frequency-domain representations map time series into distinct frequency components, providing an effective way to characterize global periodic structures and their dependencies. Among Fourier-based methods, FEDformer(Zhou et al., 2022) combines seasonal–trend decomposition with frequency-enhanced modeling and reduces computational overhead by randomly selecting a subset of Fourier modes. FreTS(Yi et al., 2023) uses multilayer perceptrons (MLPs) in the complex frequency domain to learn channel and temporal dependencies separately. FITS(Xu et al., 2024) achieves time series forecasting with a lightweight architecture through low-pass truncation and interpolation in the complex frequency domain. Beyond Fourier representations, wavelet-domain representations preserve both temporal localization and frequency-scale information through multiresolution decomposition, offering another way to characterize local variations in nonstationary time series. WaveForM(Yang et al., 2023) uses the discrete wavelet transform to decompose multivariate time series into wavelet coefficients at different levels. It combines dilated convolutions and graph convolutions to learn temporal dependencies and relationships between variables, predicting coefficients at each level. WPMixer(Murad et al., 2025) models approximation and detail coefficients in branches at different resolutions, combining local pattern extraction with global information exchange. By modeling components at different scales separately, these methods provide a multiscale representation for jointly characterizing slow variations and local details.

2.2 Temporal Representation for Time Series Forecasting

Time-domain representations extract temporal dependencies directly from historical observations and their local segments, providing a basis for characterizing local variations, evolving trends, and recent states. To separate and organize different temporal patterns, recent studies have improved these representations through time series decomposition, patch-based modeling, and multiscale fusion. Stitsyuk and Choi proposed xPatch (Stitsyuk & Choi, 2025), which uses an exponential moving average to separate trend and seasonal components. These components are modeled by an MLP-based linear branch and a convolutional neural network (CNN)-based nonlinear branch, respectively. Hu et al. proposed AMD (Hu et al., 2025a), which decomposes and aggregates temporal patterns at different scales through a multiscale decomposable mixing module. It combines temporal and channel dependency modeling with adaptive fusion of multiple predictors to capture the joint effects of multiscale patterns (Stitsyuk & Choi, 2025; Hu et al., 2025a). Beyond representation scales, the selective use of historical information has also received attention. Ma et al. proposed TimePro (Ma et al., 2025) to address how different variables influence the forecasting target at different time lags. It constructs a variable- and time-aware hyper-state that preserves fine-grained temporal features within variable tokens while adaptively selecting key time points to update the state. Hu et al. proposed TimeFilter (Hu et al., 2025b), which constructs a spatiotemporal graph from the input sequence. It filters relationships at the patch level, removing irrelevant connections while retaining key dependencies.

3 Methodology

We study multi-step forecasting for multivariate time series. For a batch of samples, let L denote the input lookback window length and C the number of variables. The historical observation sequence is represented as $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L] \in \mathbb{R}^{B\times L\times C}$, where B denotes the batch size and $\boldsymbol{x}_t \in \mathbb{R}^{B\times C}$ denotes the multivariate observations at time step t. Given a forecasting horizon H, the corresponding ground-truth future sequence is denoted by $Y = [\boldsymbol{x}_{L+1}, \ldots, \boldsymbol{x}_{L+H}] \in \mathbb{R}^{B\times H\times C}$. The forecasting model $f_\Theta(\cdot)$ takes the historical sequence X as input and generates predictions for the next H time steps, $\widehat{Y} = f_\Theta(X) \in \mathbb{R}^{B\times H\times C}$. The prediction tensor has the same shape as the ground-truth future sequence Y.

3.1 Structure Overview

To address complex time series in cloud resource load forecasting, we propose TiFuse, a selective and stable time–frequency framework for cloud workload forecasting. As shown in Figure 2, instance normalization mitigates distribution shifts across input windows. The normalized sequence

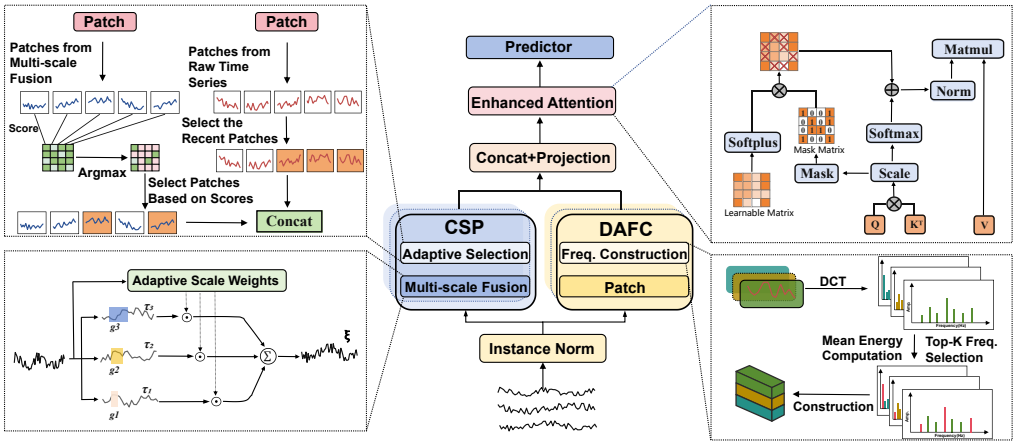


Figure 2: Overview of TiFuse. The central pipeline combines CSP and DAFC representations through concatenation and projection, followed by relation-enhanced attention and prediction. The surrounding panels illustrate adaptive patch selection, multiscale fusion, relation-enhanced attention (labeled Enhanced Attention), and frequency construction.

is divided into patches and fed into both the frequency-domain branch, DAFC, and the time-domain branch, CSP. The representations produced by the two branches are concatenated and projected, then passed through a relation-enhanced attention encoder and a predictor to forecast resource loads over multiple future time steps.

DAFC uses the DCT to construct frequency representations. Drawing on its energy compaction properties and inspired by Parseval's theorem, DAFC reduces information loss from frequency selection under a limited frequency budget, producing compact frequency-domain representations. Rather than passing all time-domain information to the model for learning, CSP combines multi-scale fusion with independent scoring across channels. It selects representative workload patterns from historical patches while filtering out patches affected by shifts or anomalies, thereby constructing an efficient time-domain representation space. Finally, relation-enhanced attention builds on the sample-specific dynamic relationships generated by standard dot-product attention. It introduces a global relationship matrix learned jointly from all training samples to provide more stable fusion of heterogeneous time-domain and frequency-domain information.

3.2 DCT-Based Adaptive Frequency Construction

Theorem 1 (Parseval's theorem)**.** For a discrete signal $y \in \mathbb{R}^L$ and its orthonormal discrete cosine transform (DCT) $Y \in \mathbb{R}^L$, the following energy identity holds:

$$\sum_{t=0}^{L-1} |y(t)|^2 = \sum_{k=0}^{L-1} |Y[k]|^2. \tag{1}$$

Theorem 1 (Strang, 1999) states that the orthonormal DCT preserves the total signal energy. The squared DCT coefficients can therefore quantify the energy contribution of each frequency component.

Reducing information loss from frequency selection under a limited frequency budget is essential for constructing compact frequency-domain representations. To this end, DAFC uses the DCT to construct real-valued frequency-domain representations and adopts retained spectral energy as the frequency selection criterion. Corollary 1 follows from Theorem 1, as detailed in Appendix A. Under a fixed frequency budget, selecting the K frequency columns with the highest average energy from all frequencies maximizes retained spectral energy. This provides a theoretical basis for the frequency selection criterion used by DAFC.

To illustrate the computation in DAFC, we first consider a single-channel historical sequence $x \in \mathbb{R}^{1\times L}$ processed by Reversible Instance Normalization (RevIN) (Kim et al., 2022). Here, L denotes the input sequence length. The model uses a time-series patch representation, following

the patching approach described in PatchTST (Nie et al., 2023). Specifically, overlapping sliding windows of length p and stride s partition x along the time axis, yielding the patch representation $P = [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_M] \in \mathbb{R}^{p \times M}$. DAFC then applies the orthonormal DCT and selects K dominant frequency columns from the full frequency representation. The overall process is expressed as

$$Z_f = \mathrm{Gather}(\mathrm{DCT}(P), \boldsymbol{\kappa}) \in \mathbb{R}^{p \times K}, \tag{2}$$

where $\boldsymbol{\kappa} = (\kappa_0, \kappa_1, \ldots, \kappa_{K-1})$ denotes the ordered dominant frequency indices determined by the current input, and Z_f is the gathered dominant frequency representation. DAFC does not perform an inverse DCT. Instead, it retains the gathered DCT coefficients for subsequent feature mapping and encoding. Specifically, the coefficient at frequency index q is defined using the orthogonal cosine basis of the DCT-II (Strang, 1999) as

$$Z_{u,q} = \alpha_q \sum_{m=0}^{M-1} P_{u,m} \cos\left[\frac{\pi}{M}\left(m + \frac{1}{2}\right) q\right], \ \alpha_q = \begin{cases} 1/\sqrt{M}, & q = 0, \\ \sqrt{2/M}, & q > 0, \end{cases} \quad q = 0, \ldots, M-1. \tag{3}$$

Following Corollary 1, the average energy at frequency index q is defined as

$$E_q = \frac{1}{p} \sum_{u=1}^{p} |Z_{u,q}|^2, \qquad q = 0, \ldots, M-1. \tag{4}$$

DAFC selects the K frequency columns with the highest average energy from all frequencies. Their ordered indices are given by

$$\boldsymbol{\kappa} = (\kappa_0, \ldots, \kappa_{K-1}) = \mathrm{ArgTopK}_K\left(\{E_q\}_{q=0}^{M-1}\right). \tag{5}$$

$\mathrm{ArgTopK}_K(\cdot)$ returns the original indices of the K frequencies with the highest energy, sorted in descending order of energy. This yields the ordered dominant frequency indices $\boldsymbol{\kappa} = (\kappa_0, \kappa_1, \ldots, \kappa_{K-1})$. Here, K denotes the fixed frequency construction budget. Across different inputs, K remains constant, whereas the specific frequency indices in $\boldsymbol{\kappa}$ adapt to the spectral energy distribution of each input. Finally, DAFC extracts the corresponding DCT components along the frequency dimension according to $\boldsymbol{\kappa}$:

$$Z_f = \mathrm{Gather}(Z, \boldsymbol{\kappa}) \in \mathbb{R}^{p \times K}, \tag{6}$$

where the frequency gathering operator $\mathrm{Gather}(\cdot)$ extracts and assembles the corresponding frequency columns from the full frequency representation Z according to the ordered indices $\boldsymbol{\kappa}$. This produces the dominant frequency representation Z_f. The selected frequencies are ordered by decreasing energy, so each frequency token position corresponds to a fixed energy rank, while its DCT frequency index may vary across inputs.

3.3 Collaboratively Selective Patches

For notational simplicity, we again describe multiscale decomposition using a single-channel time series $x \in \mathbb{R}^{1 \times L}$. The original time series contains both fine-grained local variations and coarse-grained long-term trends. Examining multiple scales helps characterize patterns at different temporal resolutions (Wang et al., 2024). CSP uses multiple average-pooling branches to extract patterns at different scales. We define the pooling kernel set as $\mathcal{G} = \{g_1, g_2, \ldots, g_R\}$, where R denotes the number of pooling scales. The current implementation uses $\mathcal{G} = \{4, 8, 12\}$. At scale r, the model first applies replication padding to preserve the sequence length after pooling. It then extracts the corresponding representation using one-dimensional average pooling with a stride of 1 and a kernel size of g_r:

$$\tau_r = \mathrm{AvgPool1d}(\mathrm{RepPad}(x); \mathrm{kernel} = g_r, \mathrm{stride} = 1), \qquad g_r \in \mathcal{G}, \quad r = 1, \ldots, R. \tag{7}$$

The output at each scale has the same length as the input sequence. Smaller pooling kernels emphasize fine-grained local variations, whereas larger kernels further smooth short-term fluctuations to capture trends over longer periods. The original input x retains fine-grained local variations, while the load patterns $\{\tau_r\}_{r=1}^{R}$ generated by the average-pooling branches describe coarse-grained information at different scales. Through a linear mapping and Softmax, the model dynamically generates position-dependent scale weights $\{\alpha_r\}_{r=1}^{R}$ from the input. It then computes a pointwise weighted sum of the trends at different scales:

$$(\alpha_1, \ldots, \alpha_R) = \mathrm{Softmax}(\mathrm{Linear}(x)), \qquad \xi = \sum_{r=1}^{R} \alpha_r \odot \tau_r \in \mathbb{R}^{1 \times L}. \tag{8}$$

Here, $\odot$ denotes elementwise multiplication, and ξ denotes the final fused scale representation. This mechanism provides learnable weights for adaptively combining fine- and coarse-grained information at different positions. Once ξ is obtained, an overlapping patching strategy partitions it along the temporal axis using a sliding window of length p and stride s. This produces M candidate load-pattern patches $\{\boldsymbol{p}_1^t, \boldsymbol{p}_2^t, \ldots, \boldsymbol{p}_M^t\}$. Each candidate patch describes a local temporal pattern, and all candidate patches are arranged chronologically to form the candidate representation P^t. To extract representative segments from the full history for forecasting, we design a multichannel selective scoring mechanism that constructs a compact historical representation under a specified selection budget. Selective scoring is used to extract these representative segments. Specifically, a multilayer perceptron (MLP) scorer assigns each candidate patch a score for each of the S selection channels. This yields a score matrix $A = \mathrm{Scorer}(P^t) \in \mathbb{R}^{M\times S}$, where A_{ij} is the score assigned to candidate patch i by selection channel j. To optimize the scorer end to end while retaining discrete selection, CSP introduces a score-based connection mechanism using a stop-gradient operator. Each selection channel extracts the highest-scoring patch, and a score normalization factor establishes a surrogate gradient path:

$$k_j = \arg\max_i A_{ij}, \qquad e_j = \frac{A_{k_j j}}{\mathrm{sg}(A_{k_j j})}, \qquad P_s = \mathrm{Stack}_{j=1}^{S}\left(e_j \boldsymbol{p}_{k_j}^t\right). \tag{9}$$

Here, $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator. For nonzero scores, e_j equals 1 in the forward pass and provides a surrogate gradient to the scorer in the backward pass. For zero scores, the product of the score and the patch is retained directly. Each channel independently determines its selection position, allowing the same candidate patch to be selected more than once.

To preserve local levels and short-term variations near the forecasting boundary, CSP applies the same sliding-window patching to the sequence after backbone preprocessing. It retains the last N_r patches in chronological order to form the recent representation P_r. Specifically, the sequence is normalized using reversible instance normalization (RevIN). If preliminary decomposition is enabled, the residual from this decomposition is taken first and then normalized. The recent branch bypasses the multiscale trend fusion and selective scorer described above to preserve recent local dynamics. By contrast, the selection branch constructs the fused trend from input x before the backbone's preliminary decomposition and RevIN. Finally, the recent representation P_r and selective representation P_s are concatenated along the patch dimension to obtain the final time-domain patch representation:

$$P_t = \mathrm{Concat}\left(P_r, P_s\right). \tag{10}$$

Here, P_r characterizes the recent state, and P_s supplements it with representative load patterns drawn from the full history and driven by the forecasting objective. The resulting P_t is the final time-domain patch representation.

3.4 Relation-Enhanced Attention

After DAFC and CSP construct frequency-domain and time-domain representations, respectively, we examine whether standard dot-product attention can capture both within-domain structure and cross-domain relationships in a unified feature space. We also consider how to improve the stability of relationship learning when processing these heterogeneous time–frequency signals. Existing adaptive graph learning methods jointly model a global graph learned from training data and an individual graph constructed from current sample features (Shi et al., 2020). Graphormer incorporates relational information into attention computation through structural encodings (Ying et al., 2021). Inspired by these approaches, we propose Relation-Enhanced Attention. Standard scaled dot-product attention (Vaswani et al., 2017) is defined as

$$\mathrm{Attn}(Q, K, V) = \mathrm{Softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d}}\right)V. \tag{11}$$

The attention matrix produced by Softmax in Eq. (11) is denoted by A. It is determined by the query–key similarities of the current sample and describes input-dependent instantaneous dependencies. To supplement these dependencies with stable structure across samples, we introduce a learnable matrix $B \in \mathbb{R}^{N_t\times N_t}$ for each attention head in each encoder layer. This matrix is shared across samples and variables. It represents shared interaction preferences between token positions in the constructed representation. Frequency positions follow energy rank, whereas temporal positions follow recent-history order or selection-channel order; corresponding positions across samples need not represent

identical frequencies or historical timestamps. It is mapped to a nonnegative relational prior through $\mathrm{Softplus}(B)$, contributing to attention enhancement by increasing connection strengths. The shared prior is not applied indiscriminately as a dense matrix over the entire relationship space. Instead, it is constrained by the dynamic correlations of the current input. Using the dot-product scores before normalization in Eq. (11), we adaptively retain connections with higher correlations. We construct an input-dependent binary sparse mask $M \in \{0,1\}^{N_t \times N_t}$ while preserving basic self-connections. The mask serves only to select prior connections. Following this sparse selection, enhanced attention is defined as

$$\mathrm{EnhAttn}(Q,K,V) = \mathrm{Norm}\left[\mathrm{Softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d}}\right) + M \odot \mathrm{Softplus}(B)\right] V. \tag{12}$$

Equation (12) preserves the responsiveness of standard attention to the current state while introducing a global relationship matrix learned jointly from all training samples. This matrix learns shared interaction preferences between token positions within and across domains. The input-dependent mask restricts the prior to candidate edges supported by the current dot-product relationships.

To examine how relation-enhanced attention affects relationship representations, Figure 3 compares the matrix structures of standard and enhanced attention. On some datasets, standard attention exhibits concentrated connections, repeated block patterns, or increasingly similar weights. Enhanced attention exhibits more differentiated connection structures and consistently higher matrix ranks, indicating richer attention representations. Complete results are provided in Figure 4 in Appendix C.

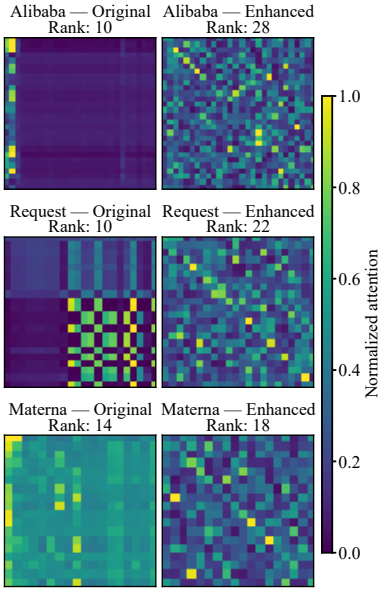


Figure 3: Matrix visualization comparison of original attention and relation-enhanced attention.

4 Experiments

4.1 Experimental Details

Datasets. To evaluate the generalization capability of the proposed model across different types of cloud resource workload forecasting tasks, we selected multiple real-world workload trace datasets: the Google cluster usage trace dataset (Wilkes, 2020), the Alibaba 2018 cluster trace dataset (Guo et al., 2019), the Alibaba 2020 cluster trace dataset (Weng et al., 2022), AcmeTrace, publicly released by the Shanghai Artificial Intelligence Laboratory (Hu et al., 2024), and the GWA-T-13 Materna workload trace dataset. Detailed dataset descriptions are provided in Appendix B.

Baseline Models. We selected eight representative time-series forecasting models as baselines: Fredformer (Piao et al., 2024), FP-MLP (Lu et al., 2025), Informer (Zhou et al., 2021), PatchTST (Nie et al., 2023), PatchMixer (Gong et al., 2023), MixLinear (Ma et al., 2026), FusAD (Zhang et al., 2026), and DTAF (Lu et al., 2026). These baselines cover joint time–frequency modeling, patch-based modeling, and specialized resource workload forecasting. MixLinear, DTAF, FusAD, and Fredformer perform joint time–frequency modeling using FFT or wavelet transforms. PatchTST and PatchMixer model local temporal segments through patches, whereas FP-MLP is a recent model specifically designed for resource workload forecasting.

4.2 Main Results

We conducted the main experiments on CPU, memory, and GPU forecasting tasks to evaluate the effectiveness of the proposed model for cloud resource workload forecasting. Following the experimental settings above, the look-back window length L was fixed at 720 for all models, and the forecasting horizon H was set to 1, 5, 10, and 30. The average forecasting results for the CPU, memory, and GPU forecasting tasks are reported in Tables 1, 2, and 3, respectively. The best results are shown in **red**, and the second-best results are underlined; ties at the reported precision receive

the same marking. Complete results for individual forecasting horizons are provided in Tables 6, 7, and 8 in Appendix B.1.

The proposed model achieved the best or second-best performance across most datasets, forecasting horizons, and evaluation metrics. Compared with the state-of-the-art joint time–frequency methods Fredformer, MixLinear, FusAD, and DTAF, it achieved lower average MSE and MAE on all datasets across the CPU, memory, and GPU tasks. It also achieved lower average MSE and MAE on most datasets than FP-MLP, a specialized resource workload forecasting model. The proposed model likewise maintained an advantage over the patch-based baselines PatchTST and PatchMixer.

Table 1: Average CPU forecasting results across datasets for forecasting horizons $H \in \{1, 5, 10, 30\}$. Full results are available in Table 6 of Appendix B.1.

Models	**TiFuse (Ours)**		Fredformer		FP-MLP		Informer		PatchTST		PatchMixer		MixLinear		FusAD		DTAF	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Usage	**0.0159**	**0.0709**	0.0244	0.1050	0.0169	0.0711	0.6770	0.7022	0.0214	0.0944	0.0370	0.1360	0.0205	0.0837	0.0261	0.0932	0.0231	0.0939
Request	**0.0128**	**0.0673**	0.0146	0.0764	0.0133	0.0699	0.4425	0.4712	0.0171	0.0913	0.0244	0.1201	0.0170	0.0866	0.0181	0.0861	0.0177	0.0885
Alibaba	**0.5005**	**0.3737**	0.5595	0.4477	0.5318	0.3782	0.9402	0.5466	0.5878	0.4628	0.5931	0.4496	0.6078	0.4559	0.5789	0.4486	0.5629	0.4284
Google2019	**0.0226**	**0.1032**	0.0259	0.1153	0.0233	0.1048	0.2602	0.4121	0.0362	0.1454	0.0325	0.1371	0.0301	0.1307	0.0239	0.1079	0.0264	0.1151
Materna	**0.0010**	**0.0132**	0.0012	0.0175	0.0014	0.0138	0.0055	0.0485	0.0016	0.0177	0.0011	0.0152	0.0012	0.0143	0.0011	0.0170	0.0014	0.0220
AcmeTrace	**0.0959**	**0.1286**	0.1031	0.1330	0.1048	0.1312	0.1300	0.1742	0.1020	0.1352	0.1027	0.1384	0.1299	0.1507	0.1100	0.1426	0.1050	0.1406

Table 2: Average memory forecasting results across datasets for forecasting horizons $H \in \{1, 5, 10, 30\}$. Full results are available in Table 7 of Appendix B.1.

Models	**TiFuse (Ours)**		Fredformer		FP-MLP		Informer		PatchTST		PatchMixer		MixLinear		FusAD		DTAF	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Usage	**0.0039**	**0.0360**	0.0069	0.0595	**0.0039**	0.0362	0.3658	0.4214	0.0075	0.0646	0.0124	0.0819	0.0043	0.0394	0.0053	0.0443	0.0078	0.0596
Request	**0.0139**	**0.0694**	0.0163	0.0840	0.0146	0.0718	0.6133	0.5810	0.0192	0.0973	0.0269	0.1214	0.0176	0.0823	0.0198	0.0884	0.0195	0.0917
Alibaba	**0.3395**	**0.3471**	0.3919	0.3950	0.3643	0.3504	0.5195	0.4671	0.4091	0.4103	0.4324	0.4168	0.4513	0.4369	0.4057	0.3926	0.3969	0.3923
Google2019	0.0814	0.1800	0.0861	0.2007	**0.0733**	**0.1737**	1.0478	0.7612	0.0903	0.2055	0.1077	0.2371	0.0965	0.2140	0.0991	0.2036	0.1020	0.2117
Materna	**0.2671**	**0.3934**	0.2780	0.4040	0.2961	0.4129	0.3828	0.4781	0.2893	0.4151	0.2827	0.4041	0.2852	0.4087	0.3006	0.4216	0.2848	0.4062
AcmeTrace	**0.0545**	0.0487	0.0603	0.0767	0.0587	**0.0465**	0.0807	0.0953	0.0558	0.0589	0.0561	0.0545	0.0787	0.0866	0.0610	0.0764	0.0551	0.0523

Table 3: Average GPU request forecasting results across datasets for forecasting horizons $H \in \{1, 5, 10, 30\}$. Full results are available in Table 8 of Appendix B.1.

Models	**TiFuse (Ours)**		Fredformer		FP-MLP		Informer		PatchTST		PatchMixer		MixLinear		FusAD		DTAF	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
MISC	**0.0090**	**0.0531**	0.0124	0.0708	0.0096	0.0554	0.3223	0.4254	0.0125	0.0752	0.0201	0.1014	0.0167	0.0880	0.0144	0.0755	0.0100	0.0581
T4	**0.0209**	**0.0784**	0.0241	0.0904	0.0214	0.0799	0.2911	0.4170	0.0275	0.1076	0.0401	0.1453	0.0416	0.1392	0.0321	0.1076	0.0226	0.0850
V100M32	**0.0077**	**0.0351**	0.0097	0.0495	0.0081	0.0377	0.4961	0.5790	0.0099	0.0527	0.0136	0.0735	0.0129	0.0643	0.0112	0.0548	0.0085	0.0421
V100	**0.0055**	0.0286	0.0066	0.0395	0.0057	**0.0281**	0.1010	0.2427	0.0107	0.0608	0.0132	0.0730	0.0107	0.0549	0.0099	0.0452	0.0059	0.0295

4.3 Ablation Study

To evaluate the effectiveness of the key components and the frequency modeling strategy, we conducted ablation experiments on multiple datasets. Results averaged across prediction horizons for each dataset are reported in Table 4, with the complete results provided in Tables 9 and 10 of Appendix B.1. To further examine the effects of the transform and frequency selection strategy in DAFC, we introduced three variants: DFT-TiFuse replaces DCT with DFT while retaining energy-based top-K frequency selection; Full-DCT retains all DCT frequency components; and Random-DCT randomly selects K DCT frequency components. The w/o CSP variant removes the selection mechanism and retains all temporal segments to assess the contribution of selecting informative historical segments. Ori-Attn replaces the enhanced attention mechanism with the original attention mechanism. The ablation results show that DAFC, CSP, and enhanced attention each contribute to predictive performance, with the complete model achieving the best average performance when all three components are combined.

Table 4: Ablation study of key components and frequency modeling strategies in TiFuse. Results are averaged over $H \in \{1, 5, 10, 30\}$. **Red**: the best results. Full results are available in Tables 9 and 10 of Appendix B.1.

Datasets	Usage		Request		Alibaba		Google2019	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
DFT-TiFuse	0.00422	0.03711	0.01614	0.07544	0.35752	0.38028	0.09561	0.22129
Full-DCT	0.00463	0.04153	0.02010	0.08617	0.41927	0.42560	0.10534	0.21036
Random-DCT	0.00409	0.03655	0.01564	0.07433	0.36708	0.36357	0.08784	0.18832
w/o CSP	0.00603	0.05321	0.02728	0.11433	0.43859	0.44000	0.09941	0.20377
Ori-Attn	0.00408	0.03695	0.01562	0.07476	0.35063	0.35612	0.08402	0.18582
TiFuse	**0.00391**	**0.03596**	**0.01392**	**0.06943**	**0.33949**	**0.34706**	**0.08142**	**0.18003**

4.4 Efficiency Analysis

Table 5 compares GPU memory consumption and training time across models. The proposed model used approximately 458–486 MiB of GPU memory across datasets. Its memory consumption was close to that of lightweight models such as MixLinear, PatchTST, and PatchMixer, and substantially lower than that of DTAF, FP-MLP, and FusAD.

The proposed model generally required less training time than most complex models. Although PatchTST and PatchMixer were faster on some datasets, the main results showed that the proposed model maintained high forecasting accuracy with good training efficiency. Overall, it achieved a favorable balance among forecasting performance, GPU memory consumption, and training time.

Table 5: Training efficiency across datasets with forecasting horizon ($H = 5$) and batch size 32, measured by training time per epoch (s/epoch) and peak GPU memory during training (MiB).

Models		Alibaba2018		Google2019		Request		Usage		Materna		AcmeTrace	
Family	Method	Memory (MiB)	Time (s/epoch)	Memory (MiB)	Time (s/epoch)	Memory (MiB)	Time (s/epoch)	Memory (MiB)	Time (s/epoch)	Memory (MiB)	Time (s/epoch)	Memory (MiB)	Time (s/epoch)
Linear	MixLinear	452	14.13	472	42.37	472	26.52	474	30.95	472	20.41	454	56.26
	TiFuse (Ours)	**482**	**15.23**	**458**	**38.68**	**480**	**20.54**	**478**	**25.83**	**480**	**16.30**	**486**	**51.92**
	Fredformer	1,068	25.37	1,836	181.30	694	43.93	1,308	76.82	870	43.68	496	40.14
Transformer	Informer	858	12.48	480	24.37	852	21.11	858	19.79	854	14.86	856	66.18
	PatchTST	458	10.19	468	22.39	462	15.26	472	16.73	470	12.01	470	33.05
	DTAF	4,534	26.74	2,038	24.93	5,492	47.34	8,962	73.32	8,310	42.31	1,162	73.34
	FP-MLP	2,044	47.19	2,884	153.10	1,892	92.66	2,308	102.78	2,516	61.89	1,524	78.55
MLP	PatchMixer	474	8.39	476	18.77	474	14.02	474	14.11	476	9.08	476	30.12
	FusAD	2,380	18.53	7,932	83.30	3,038	31.00	4,520	46.15	4,282	28.01	778	54.58

5 Conclusion

We presented TiFuse, a selective and stable time–frequency framework for cloud workload forecasting. Rather than exhaustively modeling and directly fusing complete representations from both domains, TiFuse constructs complementary information through energy-adaptive spectral selection and representative temporal patch selection, and stabilizes their interaction by combining sample-specific dependencies with a sparsely constrained global relational prior. This design explicitly controls both what each domain contributes and how the two domains interact. Across five real-world cloud traces covering CPU, memory, and GPU workloads, TiFuse achieves the lowest average MSE in 15 of 16 dataset–resource settings and the lowest average MAE in 13 of 16, while maintaining competitive memory and training efficiency. The current evaluation focuses on forecasting performance rather than downstream resource-management outcomes. Future work will investigate online forecasting and scheduling to assess how prediction improvements translate into resource utilization and service quality.

AI USE STATEMENT

Generative AI tools were used to assist with translation and language editing of the author-provided draft, LaTeX formatting, and bibliography preparation.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the source code and links to the datasets used in our experiments in the supplementary materials. The experimental settings are described in Appendix B (Experimental Details).

REFERENCES

Yisong Fu, Zezhi Shao, Chengqing Yu, Yujie Li, Zhulin An, Qi Wang, Yongjun Xu, and Fei Wang. Selective Learning for Deep Time Series Forecasting. In *NeurIPS*, volume 38, pp. 98084–98115, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/8cf54ff53f44835b9bdab2c546a1ca6d-Abstract-Conference.html.

Mohan Gao, Kexin Xu, Xiaofeng Gao, Tengwei Cai, and Haoyuan Ge. Spatial-Temporal Heterogenous Graph Contrastive Learning for Microservice Workload Prediction. *AAAI*, 39(11): 11681–11689, 2025. URL https://ojs.aaai.org/index.php/AAAI/article/view/33271.

Zeying Gong, Yujin Tang, and Junwei Liang. PatchMixer: A Patch-Mixing Architecture for Long-Term Time Series Forecasting. *arXiv preprint arXiv:2310.00655*, 2023. URL https://arxiv.org/abs/2310.00655.

Jing Guo, Zihao Chang, Sa Wang, Haiyang Ding, Yihui Feng, Liang Mao, and Yungang Bao. Who Limits the Resource Efficiency of My Datacenter: An Analysis of Alibaba Datacenter Traces. In *IWQoS*, pp. 1–10, 2019. URL https://acs.ict.ac.cn/baoyg/pub/202203/t20220317_20997.html.

Qinghao Hu, Zhisheng Ye, Zerui Wang, Guoteng Wang, Meng Zhang, Qiaoling Chen, Peng Sun, Dahua Lin, Xiaolin Wang, Yingwei Luo, Yonggang Wen, and Tianwei Zhang. Characterization of Large Language Model Development in the Datacenter. In *USENIX NSDI*, pp. 709–729, 2024. URL https://www.usenix.org/conference/nsdi24/presentation/hu.

Yifan Hu, Peiyuan Liu, Peng Zhu, Dawei Cheng, and Tao Dai. Adaptive Multi-Scale Decomposition Framework for Time Series Forecasting. In *AAAI*, volume 39, pp. 17359–17367, 2025a. URL https://ojs.aaai.org/index.php/AAAI/article/view/33908.

Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. TimeFilter: Patch-Specific Spatial-Temporal Graph Filtration for Time Series Forecasting. In *ICML*, volume 267, pp. 24893–24911, 2025b. URL https://proceedings.mlr.press/v267/hu25ac.html.

Songtao Huang, Zhen Zhao, Can Li, and Lei Bai. TimeKAN: KAN-based Frequency Decomposition Learning Architecture for Long-term Time Series Forecasting. In *ICLR*, pp. 28514–28529, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/46362971bfc3a97e6a271f2eb90fba17-Abstract-Conference.html.

Maowei Jiang, Pengyu Zeng, Kai Wang, Huan Liu, Wenbo Chen, and Haoran Liu. FECAM: Frequency enhanced channel attention mechanism for time series forecasting. *Advanced Engineering Informatics*, 58:102158, 2023. URL https://doi.org/10.1016/j.aei.2023.102158.

Dongbin Kim, Youngjoo Park, Woojin Jeong, and Jaewook Lee. Local geometry attention for time series forecasting under realistic corruptions. In *ICLR*, pp. 43205–43244, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/476ab8f369e489c04187ba84f68cfa68-Abstract-Conference.html.

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 *ICLR*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Feifei Kou, Jiahao Wang, Lei Shi, Yuhan Yao, Yawen Li, Suguo Zhu, Zhongbao Zhang, and Junping Du. CFPT: Empowering Time Series Forecasting through Cross-Frequency Interaction and Periodic-Aware Timestamp Modeling. In *ICML*, volume 267, pp. 31634–31647, 2025. URL https://proceedings.mlr.press/v267/kou25b.html.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *NeurIPS*, volume 32, pp. 5244–5254, 2019. URL https://papers.nips.cc/paper_files/paper/2019/hash/6775a0635c302542da2c32aa19d86be0-Abstract.html.

Zhenxin Li, Longquan Liao, Wenchang Zhang, Jiaying Zhang, Kaiwen Wei, Jiang Zhong, and Linjiang Zheng. SMLDR: Spectral memory learner with dual-retrieval for time series forecasting. In *IJCAI*, pp. 4527–4535, 2026. URL https://doi.org/10.24963/ijcai.2026/504.

Peiyuan Liu, Beiliang Wu, Yifan Hu, Naiqi Li, Tao Dai, Jigang Bao, and Shu-Tao Xia. TimeBridge: Non-Stationarity Matters for Long-term Time Series Forecasting. In *ICML*, volume 267, pp. 39815–39840, 2025. URL https://proceedings.mlr.press/v267/liu25cb.html.

Junkai Lu, Peng Chen, Chenjuan Guo, Yang Shu, Meng Wang, and Bin Yang. Towards Non-Stationary Time Series Forecasting with Temporal Stabilization and Frequency Differencing. *AAAI*, 40(29):24070–24078, 2026. URL https://ojs.aaai.org/index.php/AAAI/article/view/39585.

Yao Lu, Yongjing Shang, Jie Cui, Hong Zhong, Lu Liu, and Geyong Min. FP-MLP: A Frequency Domain Patch-Based MLP Model for GPU-Dominated Cloud Workload Prediction. *IEEE Transactions on Cloud Computing*, 13(4):1235–1246, 2025. URL https://doi.org/10.1109/TCC.2025.3620437.

Aitian Ma, Dongsheng Luo, and Mo Sha. MixLinear: Extreme Low Resource Multivariate Time Series Forecasting with 0.1K Parameters. In *ICLR*, pp. 77764–77789, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/7db2348b5bfeca620aa7327df815adcc-Abstract-Conference.html.

Xiaowen Ma, Zhen-Liang Ni, Shuai Xiao, and Xinghao Chen. TimePro: Efficient Multivariate Long-term Time Series Forecasting with Variable- and Time-Aware Hyper-state. In *ICML*, volume 267, pp. 42096–42111, 2025. URL https://proceedings.mlr.press/v267/ma25p.html.

Md Mahmuddun Nabi Murad, Mehmet Aktukmak, and Yasin Yilmaz. WPMixer: Efficient Multi-Resolution Mixing for Long-Term Time Series Forecasting. *AAAI*, 39(18):19581–19588, 2025. URL https://ojs.aaai.org/index.php/AAAI/article/view/34156.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *ICLR*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Xihao Piao, Zheng Chen, Taichi Murayama, Yasuko Matsubara, and Yasushi Sakurai. Fredformer: Frequency Debiased Transformer for Time Series Forecasting. In *KDD*, pp. 2400–2410, 2024. URL https://arxiv.org/abs/2406.09009.

Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-Based Action Recognition With Multi-Stream Adaptive Graph Convolutional Networks. *IEEE Transactions on Image Processing*, 29: 9532–9545, 2020. URL https://doi.org/10.1109/TIP.2020.3028207.

Artyom Stitsyuk and Jaesik Choi. xPatch: Dual-Stream Time Series Forecasting with Exponential Seasonal-Trend Decomposition. In *AAAI*, volume 39, pp. 20601–20609, 2025. URL https://ojs.aaai.org/index.php/AAAI/article/view/34270.

Gilbert Strang. The Discrete Cosine Transform. *SIAM Review*, 41(1):135–147, 1999. URL https://epubs.siam.org/doi/abs/10.1137/S0036144598336745.

Yanru Sun, Zongxia Xie, Emadeldeen Eldele, Dongyue Chen, Qinghua Hu, and Min Wu. Learning Pattern-Specific Experts for Time Series Forecasting Under Patch-level Distribution Shift. In *NeurIPS*, volume 38, pp. 91810–91844, 2025. URL https://papers.nips.cc/paper_files/paper/2025/file/8491a7fcc218946b471b600a915c8b02-Paper-Conference.pdf.

Hwa Hui Tew, Junn Yong Loo, Fang Yu Leong, Julia K. Lau, Ding Fan, Hernando Ombao, Raphaël C.-W. Phan, Chee Pin Tan, and Chee-Ming Ting. Functional MRI Time Series Generation via Wavelet-Based Image Transform and Spectral Flow Matching for Brain Disorder Identification. In *ICLR*, pp. 106373–106393, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/ad9804eed175610302917a0c21ab9b52-Abstract-Conference.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NeurIPS*, volume 30, pp. 5998–6008, 2017. URL https://papers.nips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Zhang, and Jun Zhou. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. In *ICLR*, pp. 38626–38652, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/a7ac8a21e5a27e7ab31a5f42a0117bdb-Abstract-Conference.html.

Qizhen Weng, Wencong Xiao, Yinghao Yu, Wei Wang, Cheng Wang, Jian He, Yong Li, Liping Zhang, Wei Lin, and Yu Ding. MLaaS in the Wild: Workload Analysis and Scheduling in Large-Scale Heterogeneous GPU Clusters. In *USENIX NSDI*, pp. 945–960, 2022. URL https://www.usenix.org/conference/nsdi22/presentation/weng.

John Wilkes. Google Cluster-Usage Traces v3. Technical report, Google Inc., 2020. URL https://github.com/google/cluster-data/blob/master/ClusterData2019.md.

Zhijian Xu, Ailing Zeng, and Qiang Xu. FITS: Modeling Time Series with 10k Parameters. In *ICLR*, pp. 26295–26318, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/701251e1db4a2e4dd2ef23f5265d5936-Abstract-Conference.html.

Fuhao Yang, Xin Li, Min Wang, Hongyu Zang, Wei Pang, and Mingzhong Wang. WaveForM: Graph Enhanced Wavelet Learning for Long Sequence Forecasting of Multivariate Time Series. *AAAI*, 37(9):10754–10761, 2023. URL https://ojs.aaai.org/index.php/AAAI/article/view/26276.

Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain MLPs are More Effective Learners in Time Series Forecasting. In *NeurIPS*, volume 36, pp. 76656–76679, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/hash/f1d16af76939f476b5f040fd1398c0a3-Abstract-Conference.html.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do Transformers Really Perform Badly for Graph Representation? In *NeurIPS*, volume 34, pp. 28877–28888, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f1c1592588411002af340cbaedd6fc33-Abstract.html.

Da Zhang, Bingyu Li, Zhiyuan Zhao, Feiping Nie, Junyu Gao, and Xuelong Li. FusAD: Time-Frequency Fusion with Adaptive Denoising for General Time Series Analysis. In *ICDE*, pp. 1334–1348, 2026. URL https://arxiv.org/abs/2512.14078.

Xiang Zhang, Ziyuan Zhao, Theodoros Tsiligkaridis, and Marinka Zitnik. Self-Supervised Contrastive Pre-Training for Time Series via Time-Frequency Consistency. In *NeurIPS*, volume 35, pp. 3988–4003, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/194b8dac525581c346e30a2cebe9a369-Abstract-Conference.html.

Feiyu Zhao, Weiwei Lin, Shengsheng Lin, Shaomin Tang, and Keqin Li. MSCNet: Multi-Scale Network With Convolutions for Long-Term Cloud Workload Prediction. *IEEE Transactions on Services Computing*, 18(2):969–982, 2025. URL https://ieeexplore.ieee.org/document/10858411/.

Haowen Zheng, Yao Lu, Zekun Sun, John Panneerselvam, Xiang Sun, and Lu Liu. Energy Optimisation in Cloud Datacentres with MC-TIDE: Mixed Channel Time-series Dense Encoder for Workload Forecasting. *Applied Energy*, 374:123903, 2024.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *AAAI*, 35 (12):11106–11115, 2021. URL https://doi.org/10.1609/aaai.v35i12.17325.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In *ICML*, volume 162, pp. 27268–27286, 2022. URL https://proceedings.mlr.press/v162/zhou22g.html.