SCPEC: STATE-CONDITIONED PATTERN-EXPERT COMPOSITION FOR WORKLOAD FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Workload forecasting in dynamic multi-tenant edge-cloud platforms is crucial for resource provisioning, application deployment, and service assurance. However, heterogeneity across nodes and workload dynamics within each node make accurate forecasting challenging. Recent studies have adopted unified forecasting models to handle both, but representative ones still rely on a prototype pool of stored historical patterns, which limits their generalization to evolving workloads. To address this limitation, we reformulate multi-node workload forecasting as a multi-view regression problem, in which each node's workload state is jointly characterized through temporal, spectral, and static-context views, and reusable knowledge is selected conditioned on this state. Building on this formulation, we propose SCPEC, which replaces the historical prototype pool with state-conditioned composition of learnable pattern experts, thereby improving both generalization and forecasting accuracy. Experiments on eight real-world workload, electricity, and traffic datasets show that SCPEC achieves the best or tied-best results on 15 of 16 MSE/MAE evaluation entries. Our code is publicly available at `https://anonymous.4open.science/r/test-F170/`.

1 INTRODUCTION

Large-scale multi-tenant edge-cloud platforms host numerous dynamic applications on heterogeneous, resource-constrained nodes, where accurate workload forecasting underpins proactive resource provisioning, application deployment, and service-level assurance (Huang et al., 2023; Qu et al., 2018; Kaviani et al., 2026). Because node-specific predictors are costly to train, deploy, and maintain at this scale, practical platforms favor serving all nodes with a single unified forecasting model (Huang et al., 2023; 2026). However, heterogeneity across nodes and workload dynamics within each node make accurate unified forecasting challenging. Nodes differ in resource capacity, network conditions, and hosted applications, while each node's workload is continually reshaped by new-application arrival, application switching, and workload-state transitions (Fig. 1; Huang et al., 2023; 2025). When a new application is deployed, for instance, provisioning must anticipate a load pattern the node has never exhibited. A unified model must therefore carry knowledge that is reusable across heterogeneous nodes and remains valid as each node's workload evolves.

Recent unified forecasters share such knowledge across nodes (Huang et al., 2023; Li et al., 2025b; Huang et al., 2025): DynEformer extracts global application patterns and adds static node context (Huang et al., 2023; 2026), and MetaEformer refines them into meta-patterns maintained and reused through Meta-pattern Pooling and Echo (Huang et al., 2025). Both, however, draw this knowledge from a prototype pool of patterns extracted from observed history, so even what they compose is built from observed patterns; MetaEformer further selects from its pool by similarity to the current window, which is unreliable when similar histories lead to different futures. Neighboring work strengthens structural modeling with frequency or primitive representations (Ye et al., 2025; Wu et al., 2026), or adaptation to non-stationarity with stationarity-aware retrieval and drift-aware expert routing (Zhou et al., 2026; Zhu et al., 2026), but does not jointly address how shared pattern knowledge can reach beyond a history-derived pool and be selected when application events change its applicability. This calls for reusable knowledge that is not restricted to observed history and is reused according to the workload state rather than waveform similarity alone.

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.

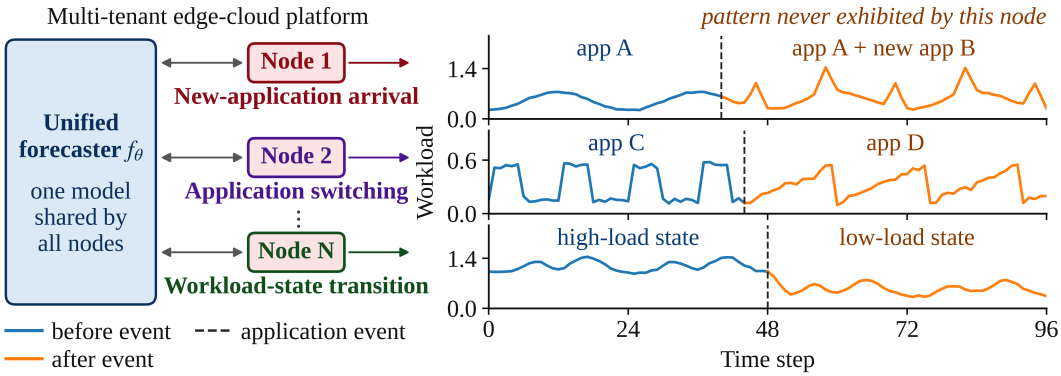


Figure 1: Multi-node edge-cloud workload forecasting: one unified forecaster serves heterogeneous nodes whose workloads change after application events (dashed lines).

In response, we reformulate unified multi-node workload forecasting as a multi-view regression problem. Our key idea is to characterize each node's workload state jointly through temporal, spectral, and static-context views, and to compose reusable knowledge conditioned on this state rather than retrieve it from a finite pool of historical patterns. This changes two things at once: reusable knowledge becomes a learnable object rather than an index over observed history, and its selection depends on the workload state rather than on waveform resemblance. Our goal is to learn state-conditioned knowledge that generalizes to workloads not observed during training.

However, heterogeneous and dynamic workloads present two challenges. We quantify both on the real-world edge-cloud workload dataset ECW and its two application-event scenarios, NewAPP (new-application arrival) and Switch (application switching) (Huang et al., 2023; 2025); models trained on ECW are also evaluated on the latter two without retraining. **First, reusable knowledge must reach beyond observed history while remaining shared across nodes.** Post-event workloads increasingly lack a close historical match: with ECW training history as the reference, the fraction of queries that retain a high-fidelity historical match (similarity $\rho \geq 0.95$) falls from 85.3% on ECW to 66.3% on NewAPP (Fig. 2(a)), a *history-bound coverage bottleneck*. Yet reaching beyond history must not forfeit the explicit, pattern-level reuse across heterogeneous nodes that motivated prototype pools in the first place (Huang et al., 2023; 2025).

Second, which knowledge applies depends on a state that the local waveform does not reveal. Application events shift the workload distribution (Fig. 2(b1)) and, more critically, alter how history maps to the future. Similar histories need not share similar futures under non-stationarity (Zhou et al., 2026), and application events make this far more frequent: among queries that still retain a highly similar historical match ($\rho \geq 0.90$), the fraction whose futures diverge rises from 2.81% on ECW to 17.77% on NewAPP (Fig. 2(b2)), a $6.32\times$ amplification, still $3.8\times$ in a historical-composition diagnostic, that we term an *application-conditioned applicability shift*. The governing state is reflected jointly in recent dynamics, global periodicity (Ye et al., 2025), and node context (Huang et al., 2023; 2026), which differ in form and time scale.

To address these challenges, we propose **SCPEC** (State-Conditioned Pattern-Expert Composition). For the first challenge, SCPEC replaces the history-derived prototype pool with a learnable pattern-expert bank, jointly optimized with the forecaster and shared across nodes, so that reusable knowledge stays explicit and pattern-level without being indexed by observed history.

To handle the application-conditioned applicability shift, SCPEC fuses the three views into a workload state: temporal tokens encode recent evolution, a global spectral descriptor is adaptively gated into each token, and, when available, static node attributes supply operating context. A sparse Top-K router then selects experts from this state rather than by waveform similarity, following the conditional-computation principle of Mixture-of-Experts (Shazeer et al., 2017). Unlike Mixture-of-Experts forecasters (Liu, 2025; Zhu et al., 2026), whose experts are forecasting subnetworks, the experts in SCPEC are the reusable pattern knowledge itself.

We make the following contributions:

- **Problem formulation.** We reformulate unified multi-node workload forecasting as a multi-view regression problem in which reusable knowledge is selected conditioned on a temporal–spectral–static-context workload state, and we identify and empirically characterize the two challenges this formulation must meet under dynamic application states.
- **SCPEC.** We propose SCPEC, which replaces the historical prototype pool with a learnable pattern-expert bank that learns reusable patterns jointly with the forecasting objective while keeping them shared across nodes. It selects experts from this bank through sparse routing conditioned on the multi-view workload state rather than on waveform similarity.
- **Extensive evaluation.** On eight real-world workload, electricity, and traffic datasets against 20 baselines, SCPEC achieves the best or tied-best result on 15 of 16 MSE/MAE entries. Relative to MetaEformer, the most closely related prototype-pool method (our run of its official code), it reduces MSE by 8.82% on ECW and by 9.38% and 12.90% on NewAPP and Switch without retraining (Fig. 2(c)).

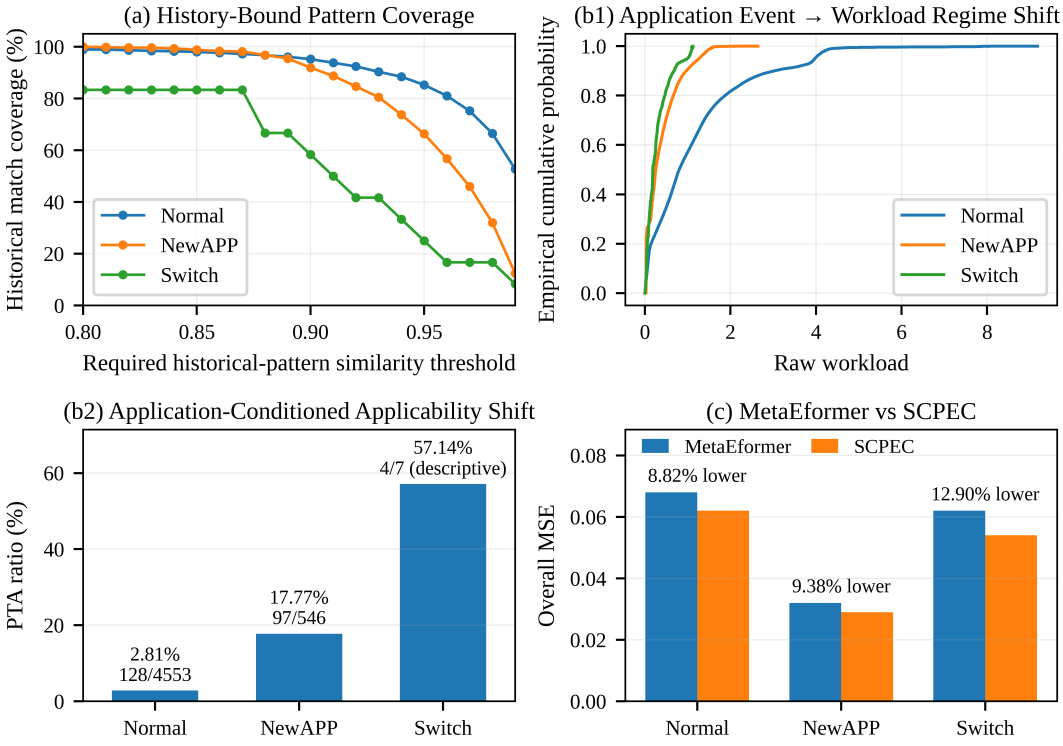


Figure 2: Motivating analysis on ECW (labeled Normal) and its application-event scenarios NewAPP and Switch (Sec. 3.2); historical matches come from the ECW training split only. (a) History-bound coverage bottleneck: fraction of queries whose best historical match reaches similarity ρ. (b1) Empirical CDFs of raw workloads. (b2) Application-conditioned applicability shift: PTA ratio among queries covered at $\rho = 0.90$. (c) MSE of SCPEC and our run of MetaEformer, trained on ECW only and evaluated on NewAPP and Switch without retraining.

2 Related Work

Cloud and edge workload forecasting has evolved from correlation-aware and structure-specific modeling toward shared and reusable workload representations. Early collaborative approaches exploit cross-site workload correlations or time–frequency and multi-scale structures (Li et al., 2023; 2025a; Zhao et al., 2025b;a). For dynamic multi-tenant edge-cloud platforms, DynEformer introduces unified forecasting with global application patterns and static context (Huang et al., 2023; 2026), while Scout explores collaborative representations across heterogeneous tenants (Li et al., 2025b). MetaEformer advances this direction by maintaining and reusing fine-grained meta-patterns

through Meta-pattern Pooling and Echo (Huang et al., 2025). Recent workload-specific methods additionally exploit frequency-domain structures or reusable workload primitives (Ye et al., 2025; Wu et al., 2026). Despite these advances, how the historical coverage and predictive applicability of reusable pattern knowledge change with application states remains insufficiently characterized.

Another line of research addresses retrieval reliability and adaptive expert selection under heterogeneous or non-stationary time series. SARAF shows that highly similar historical segments do not necessarily imply similar futures and improves retrieval with stationarity-aware mechanisms (Zhou et al., 2026). Meanwhile, recent Mixture-of-Experts forecasters specialize experts according to frequency structure, temporal regimes, distribution shifts, or richer state descriptors (Liu, 2025; Liu et al., 2025; Zhu et al., 2026; Hong et al., 2026; Ortigossa et al., 2026; Li et al., 2026; Wang et al., 2026). These studies show that retrieval reliability and expert suitability are state dependent, but they target generic non-stationarity or forecasting-expert adaptation. We instead study how pattern knowledge shared across nodes should be represented and selected when application events weaken both its historical coverage and its predictive applicability.

3 Problem Formulation and Empirical Insights

3.1 Unified Dynamic Workload Forecasting as Multi-View Regression

Consider N heterogeneous nodes indexed by i. At time t, let $\mathbf{x}^{(i)}_{t-L+1:t} \in \mathbb{R}^L$ be the workload of node i over a look-back window of length L, with exogenous temporal covariates $\mathbf{z}^{(i)}_{t-L+1:t} \in \mathbb{R}^{L\times d_z}$ (e.g., hour-of-day) and an optional static node context $\mathbf{s}^{(i)} \in \mathbb{R}^{d_s}$ (e.g., resource capacity) (Wang et al., 2024). A unified forecaster f_θ shared by all nodes (Huang et al., 2023) predicts the next H steps. We formulate it as a regression on a multi-view workload state, with reusable knowledge selected conditioned on that state:

$$\hat{\mathbf{x}}^{(i)}_{t+1:t+H} = f_\theta\big(\mathbf{x}^{(i)}_{t-L+1:t}, \mathbf{z}^{(i)}_{t-L+1:t}, \mathbf{s}^{(i)}\big) = g_\theta\big(\mathbf{u},\, \mathcal{C}(\boldsymbol{\alpha}(\mathbf{u}), \mathcal{P})\big), \tag{1}$$

where the workload state $\mathbf{u} = \Psi(\mathbf{v}^{\mathrm{tem}}, \mathbf{v}^{\mathrm{spec}}, \mathbf{v}^{\mathrm{ctx}})$ fuses complementary views of recent dynamics from $(\mathbf{x}, \mathbf{z})$, the global periodicity of $\mathbf{x}$, and the static context $\mathbf{s}$, which differ in form and time scale; $\mathcal{P} = \{\mathbf{p}_e\}_{e=1}^{E}$ is a reusable pattern source shared by all nodes, $\boldsymbol{\alpha}(\mathbf{u})$ are state-dependent selection weights, and $\mathcal{C}$ composes the selected patterns. f_θ is trained by minimizing the mean squared H-step error $\mathcal{L}_{\mathrm{pred}}$ over the node–time training windows $(i,t) \in \mathcal{D}_{\mathrm{train}}$. The setting is dynamic: after training, new-application arrival, application switching, and workload-state transitions (Fig. 1) change the workload distribution and, more critically, how history maps to the future. Our application-event protocol instantiates the first two: models trained on ECW are evaluated on NewAPP and Switch without retraining. In Fig. 3, $\mathbf{u}$ is $\mathbf{H}^{(0)}$ built from $\mathbf{T}$, $\mathbf{f}$, and $\phi_s(\mathbf{s})$; steps (1)–(3) of $\mathcal{E}^{(\ell)}$ realize $\boldsymbol{\alpha}$, $\mathcal{P}$, and $\mathcal{C}$ as sparse Top-K routing, a learnable bank, and gated recombination; and g_θ comprises the Transformer stages and the readout.

History-derived pattern pools as a restricted instantiation. Pattern-pool forecasters can be viewed as a restricted instantiation of Eq. (1) whose pattern source is constructed and maintained from observed historical waveforms rather than jointly optimized with the forecasting objective (Huang et al., 2023; 2025); MetaEformer further computes $\boldsymbol{\alpha}$ from the similarity between the current window and the pool (Huang et al., 2025). Such reuse assumes **(A1)** *support sufficiency*: observed history offers sufficiently faithful support for the windows met at deployment; similarity-based selection further assumes **(A2)** *future consistency*: historically similar windows have similar futures. Eq. (1) does not explicitly impose either assumption: its pattern source need not be an index over observed history, and $\boldsymbol{\alpha}$ may depend on the workload state rather than on waveform resemblance.

3.2 Empirical Characterization of the Two Challenges

We characterize when and how (A1) and (A2) become unreliable under application events. The reference set $\mathcal{R}$ holds every ECW training window, the complete history from which pattern pools are built and thus the most favorable single-window reference for history-bound reuse; the queries $\mathcal{Q}$ are the windows of the ECW test period, NewAPP, and Switch. Sim is the Pearson correlation of short seasonal segments, $r^\star(q) = \arg\max_{r\in\mathcal{R}} \mathrm{Sim}(q,r)$, and $\mathbf{y}_q$ is the future of q (App. B.2).

History-bound coverage bottleneck (A1). The coverage $\mathrm{Cov}(\rho)$, i.e., the fraction of queries with $\mathrm{Sim}(q, r^\star(q)) \geq \rho$, falls at $\rho = 0.95$ from 85.3% on ECW to 66.3% on NewAPP, with only 3 of 12 Switch windows covered, and the gap widens as ρ increases (Fig. 2(a)): single-window support (A1) fails precisely for post-event workloads.

Application-conditioned applicability shift (A2). Beyond shifting the workload distribution (Fig. 2(b1)), application events alter how history maps to the future. Among queries covered at $\rho = 0.90$, the pattern-transfer ambiguity (PTA) ratio, i.e., the fraction whose best match has a divergent future with $\mathrm{Corr}(\mathbf{y}_q, \mathbf{y}_{r^\star(q)}) < 0.5$, rises from 2.81% (128/4553) on ECW to 17.77% (97/546) on NewAPP, a 6.32× amplification (3.8× in the composition diagnostic of App. B.2); 4 of 7 covered Switch queries are PTA cases (Fig. 2(b2)). Similar histories need not share similar futures under non-stationarity (Zhou et al., 2026); application events make this far more frequent, so (A2) becomes unreliable: local waveform similarity alone is insufficient to reliably determine which knowledge to reuse.

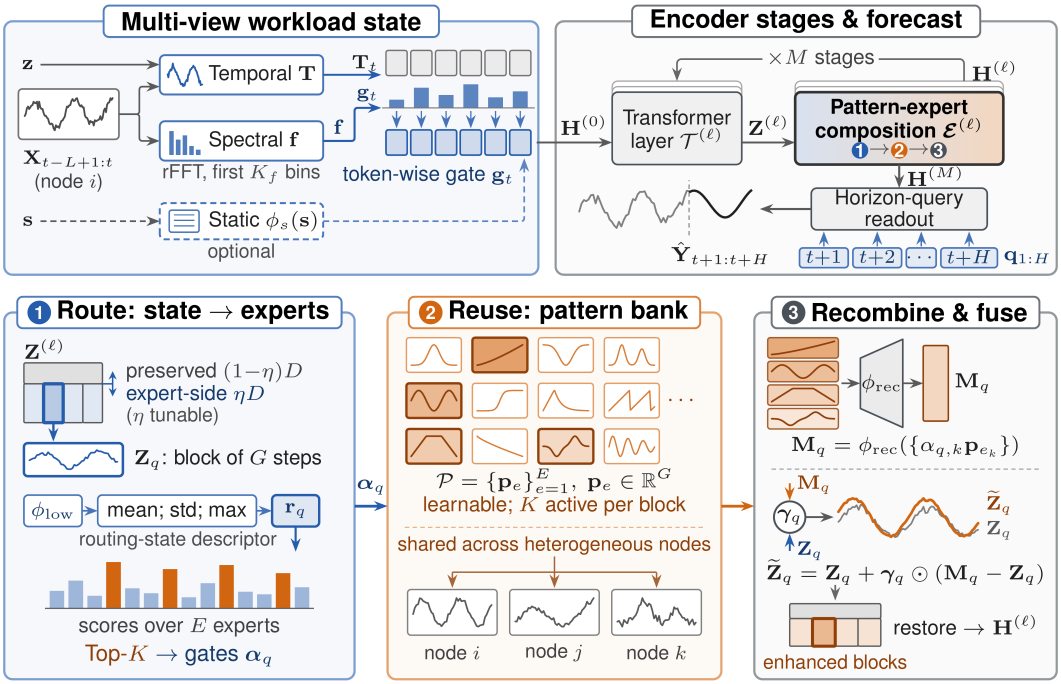


Figure 3: Overview of SCPEC. Top: the multi-view workload state, built from the look-back window $\mathbf{X}$, temporal covariates $\mathbf{z}$, and optional static context $\mathbf{s}$, feeds M interleaved Transformer / pattern-expert stages, and a horizon-query readout predicts all H steps in one pass. Bottom: inside $\mathcal{E}^{(\ell)}$, (1) each local block $\mathbf{Z}_q$ is summarized into a routing-state descriptor $\mathbf{r}_q$ that selects K of the E experts; (2) the experts are learnable length-G patterns shared by all nodes, with one bank per stage; (3) the selected patterns are recombined into $\mathbf{M}_q$ and fused back into $\mathbf{H}^{(\ell)}$ via a gated residual.

4 Method

4.1 Model Overview

To instantiate the unified forecaster f_θ of Eq. (1), we develop **SCPEC**, illustrated in Fig. 3. The look-back window is first mapped to a workload-state representation $\mathbf{H}^{(0)}$ that integrates temporal dynamics, spectral structure, and, when available, static node context (Sec. 4.2). Starting from $\mathbf{H}^{(0)}$, the model interleaves Transformer-based temporal encoding with state-conditioned sparse Top-K pattern-expert composition (Sec. 4.3). Rather than sourcing reusable knowledge from a history-derived pattern pool, SCPEC represents it with learnable pattern-expert banks shared by all nodes. A horizon-query readout then predicts all future steps directly without recursion (Sec. 4.4).

4.2 TEMPORAL–SPECTRAL–CONTEXT WORKLOAD-STATE REPRESENTATION

Since local waveform similarity alone is insufficient to reliably determine which knowledge to reuse after application events (Sec. 3.2), we characterize the current workload from temporal, spectral, and contextual views so that expert selection depends on the workload state.

Given the look-back window $\mathbf{X} \in \mathbb{R}^{L\times C}$, where we keep an explicit channel axis for implementation and $C = 1$ in all experiments, and its temporal covariates $\mathbf{z}$, we first obtain temporal tokens $\mathbf{T} \in \mathbb{R}^{L\times D}$ through value, temporal, and positional embedding. To complement local temporal dynamics with window-level spectral structure, we remove the per-channel temporal mean and apply a Hann window. Let $\mathbf{F}_{K_f} = \Pi_{K_f}\Big(\mathrm{rFFT}(\widetilde{\mathbf{X}})\Big)$ denote the fixed-length complex spectrum, where $\Pi_{K_f}(\cdot)$ retains the first K_f frequency bins when available and zero-pads the frequency dimension otherwise. We construct the spectral descriptor as

$$\mathbf{f} = \phi_f\big(\big[\mathrm{Pool}_c\,\mathrm{Re}(\mathbf{F}_{K_f}); \mathrm{Pool}_c\,\mathrm{Im}(\mathbf{F}_{K_f})\big]\big), \qquad \mathbf{f} \in \mathbb{R}^D, \tag{2}$$

where $\widetilde{\mathbf{X}}$ denotes the de-meaned and windowed workload, Pool_c aggregates channel-wise spectral information (the identity for $C = 1$), and $\phi_f(\cdot)$ is a learnable projection. We broadcast $\mathbf{f}$ along the temporal dimension as $\overline{\mathbf{F}} \in \mathbb{R}^{L\times D}$ and adaptively fuse it with the temporal representation:

$$\mathbf{H}^{tf}_t = \mathbf{T}_t + \mathbf{g}_t \odot \phi_{tf}\big([\mathbf{T}_t; \overline{\mathbf{F}}_t]\big), \qquad \mathbf{g}_t = \sigma\big(\mathbf{W}_f[\mathbf{T}_t; \overline{\mathbf{F}}_t] + \mathbf{b}_f\big), \tag{3}$$

where $\mathbf{g}_t$ is a token-wise gate. Static or slowly varying node context $\mathbf{s}$ is projected into the same hidden space and, when available, fused with the temporal–spectral representation:

$$\mathbf{H}^{(0)} = \Phi_{\mathrm{ctx}}\left(\mathbf{H}^{tf}, \phi_s(\mathbf{s})\right), \tag{4}$$

where Φ_{ctx} adds the context embedding to every token and applies layer normalization; without static context, $\mathbf{H}^{(0)} = \mathbf{H}^{tf}$. Thus, $\mathbf{H}^{(0)}$ instantiates the state $\mathbf{u}$ of Eq. (1), with $\mathbf{T}$, $\mathbf{f}$, and $\phi_s(\mathbf{s})$ as its three views and Eqs. (3)–(4) as the fusion Ψ. Details are given in App. A.1.

4.3 STATE-CONDITIONED SPARSE PATTERN-EXPERT COMPOSITION

As detailed in Fig. 3, SCPEC instantiates Eq. (1) without explicitly relying on (A1) or (A2). At each stage ℓ, it parameterizes the pattern source as a learnable pattern-expert bank $\mathcal{P}^{(\ell)} = \{\mathbf{p}^{(\ell)}_1, \ldots, \mathbf{p}^{(\ell)}_E\}$ shared by all nodes, where each $\mathbf{p}^{(\ell)}_e \in \mathbb{R}^G$ is a reusable workload pattern of length G optimized jointly with the forecasting objective, and selects experts from the workload state.

At stage ℓ, Transformer encoding first gives $\mathbf{Z}^{(\ell)} = \mathcal{T}^{(\ell)}(\mathbf{H}^{(\ell-1)})$. An application ratio η sets $D_{\mathrm{exp}} = \lfloor \eta D \rfloor$, and we split $\mathbf{Z}^{(\ell)}$ into a preserved part $\mathbf{Z}^{(\ell)}_{\mathrm{keep}}$ with $D - D_{\mathrm{exp}}$ dimensions and an expert-side part $\mathbf{Z}^{(\ell)}_{\mathrm{exp}}$ with D_{exp} dimensions. We partition $\mathbf{Z}^{(\ell)}_{\mathrm{exp}}$ into local workload blocks $\mathbf{Z}^{(\ell)}_{\mathrm{exp},q}$ of length G ($\mathbf{Z}_q$ in Fig. 3) and project each block as $\mathbf{A}^{(\ell)}_q = \phi_{\mathrm{low}}(\mathbf{Z}^{(\ell)}_{\mathrm{exp},q})$. Its routing-state descriptor $\mathbf{r}^{(\ell)}_q$ then instantiates the selection weights $\boldsymbol{\alpha}$ of Eq. (1) through sparse Top-K routing, with the statistics taken over the G positions of the block:

$$\begin{aligned} \mathbf{r}^{(\ell)}_q &= \Big[\mathrm{Mean}(\mathbf{A}^{(\ell)}_q); \mathrm{Std}(\mathbf{A}^{(\ell)}_q); \mathrm{Max}(\mathbf{A}^{(\ell)}_q)\Big], \\ \mathbf{a}^{(\ell)}_q = \mathbf{W}_r\mathbf{r}^{(\ell)}_q + \mathbf{b}_r, \qquad \mathcal{K}^{(\ell)}_q &= \mathrm{TopK}(\mathbf{a}^{(\ell)}_q, K), \qquad \boldsymbol{\alpha}^{(\ell)}_q = \mathrm{Softmax}\left(\mathbf{a}^{(\ell)}_{q,\mathcal{K}^{(\ell)}_q}\right). \end{aligned} \tag{5}$$

Because routing originates from the state-aware hidden representation, different local states activate different expert compositions; sparse Top-K selection follows the conditional-computation principle of Mixture-of-Experts (Shazeer et al., 2017), but our experts are reusable workload patterns rather than feed-forward subnetworks. App. A.2 details routing, recovery, and gating.

As in the Echo layer of MetaEformer (Huang et al., 2025), the selected experts are not collapsed into a single retrieved waveform; their weighted activations are retained along the expert dimension and recovered into the hidden space, instantiating the composition $\mathcal{C}$ (batch index suppressed):

$$\mathbf{F}^{(\ell)}_{q,j,k} = \alpha^{(\ell)}_{q,k}[\mathbf{p}^{(\ell)}_{e_k}]_j, \qquad \mathbf{M}^{(\ell)}_{q,j} = \phi_{\mathrm{rec}}\left(\mathbf{F}^{(\ell)}_{q,j,:}\right), \quad \phi_{\mathrm{rec}} : \mathbb{R}^K \to \mathbb{R}^{D_{\mathrm{exp}}}. \tag{6}$$

Multiple learnable patterns thus jointly construct reusable knowledge for the current local state without requiring a matched historical prototype. To preserve input-specific information, the recovered representation is integrated through a gated residual:

$$\widetilde{\mathbf{Z}}^{(\ell)}_{\mathrm{exp},q} = \mathbf{Z}^{(\ell)}_{\mathrm{exp},q} + \boldsymbol{\gamma}^{(\ell)}_q \odot \left(\mathbf{M}^{(\ell)}_q - \mathbf{Z}^{(\ell)}_{\mathrm{exp},q}\right), \tag{7}$$

where $\boldsymbol{\gamma}^{(\ell)}_q$ is a learnable sigmoid gate conditioned on the original and recovered representations. The enhanced blocks are restored to the sequence and combined with the preserved dimensions to obtain $\mathbf{H}^{(\ell)}$. The bank is randomly initialized rather than taken from a precomputed prototype pool. During the first N_{warm} epochs, each expert is additionally set to an exponential moving average of the input workload segments for which it is the top-ranked expert; afterwards, $\mathcal{P}^{(\ell)}$ is updated solely by gradient-based optimization of the training objective (App. A.3).

4.4 Interleaved Encoding and Multi-Horizon Forecasting

Stacking M stages interleaves pattern-expert composition with temporal encoding rather than applying it as a separate post-processing module:

$$\mathbf{Z}^{(\ell)} = \mathcal{T}^{(\ell)}\left(\mathbf{H}^{(\ell-1)}\right), \qquad \mathbf{H}^{(\ell)} = \mathcal{E}^{(\ell)}\left(\mathbf{Z}^{(\ell)}\right), \quad \ell = 1, \ldots, M, \tag{8}$$

where $\mathcal{T}^{(\ell)}$ denotes Transformer-based temporal encoding and $\mathcal{E}^{(\ell)}$ the state-conditioned pattern-expert composition of Sec. 4.3. This interleaving lets temporal representations and state-conditioned reusable knowledge evolve jointly within the same hidden space. App. A.5 gives its complexity.

For direct multi-step forecasting, we introduce H learnable horizon queries $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_H] \in \mathbb{R}^{H\times D}$. Let $\overline{\mathbf{H}} = \mathrm{LN}_{\mathrm{enc}}(\mathbf{H}^{(M)})$ denote the final normalized encoder representation. The readout is

$$\mathbf{R} = \mathrm{MHA}(\mathbf{Q}, \overline{\mathbf{H}}, \overline{\mathbf{H}});\ \mathbf{R}_1 = \mathrm{LN}_1(\mathbf{R});\ \widetilde{\mathbf{R}} = \mathrm{LN}_2\big(\mathbf{R}_1 + \mathrm{FFN}(\mathbf{R}_1)\big);\ \widehat{\mathbf{Y}} = \mathbf{W}_o\widetilde{\mathbf{R}} + \mathbf{b}_o. \tag{9}$$

Each horizon query can therefore retrieve different information from the common history representation, while all future steps are predicted jointly without recursion; with $C = 1$, $\widehat{\mathbf{Y}}$ is the forecast $\hat{\mathbf{x}}^{(i)}_{t+1:t+H}$ of Eq. (1). The Transformer stages and this readout form g_θ in Eq. (1). SCPEC is trained by minimizing $\mathcal{L} = \mathcal{L}_{\mathrm{pred}} + \lambda_{\mathrm{aux}}\mathcal{L}_{\mathrm{aux}}$, where $\mathcal{L}_{\mathrm{pred}}$ is the forecasting loss of Sec. 3.1 and $\mathcal{L}_{\mathrm{aux}}$ is an auxiliary sparse-routing regularizer (App. A.4). Algorithm 1 gives the full forward pass.

5 Experiments

5.1 Experimental Settings

Datasets. Following the benchmark setting of MetaEformer (Huang et al., 2025), we evaluate SCPEC on eight real-world datasets covering cloud workload, electricity, and traffic forecasting. CBW and ECW are cloud-workload datasets, while NewAPP and Switch are two ECW-derived application-event scenarios corresponding to new-application arrival and application switching. We further use three electricity-load subsets, ECL32, ECL57, and ECL270, together with Traffic852 to evaluate performance across application domains. For all datasets, models observe the previous $L = 48$ time steps and predict the next $H = 24$ steps. For NewAPP and Switch, models trained on ECW are evaluated without target-scenario retraining, following the zero-shot application-state transfer setting (Huang et al., 2023; 2025).

Baselines. We compare against 20 representative forecasting methods spanning different modeling paradigms. General time-series methods include UniTime, PaiFilter, FreTS, TSMixer, Crossformer, TimesNet, DLinear, Quatformer, FEDformer, and Autoformer (Liu et al., 2024; Yi et al., 2024; 2023; Chen et al., 2023; Zhang & Yan, 2023; Wu et al., 2023; Zeng et al., 2023; Chen et al., 2022; Zhou et al., 2022; Wu et al., 2021). We further include non-stationarity-aware methods SAN, Koopa, and Non-stationary Transformer (Liu et al., 2023b;a; 2022), as well as workload- and data-pooling-oriented methods DynEformer, VaRDE-L, MetaEformer, and Fremer (Huang et al., 2023; Yang & Youn, 2023; Huang et al., 2025; Ye et al., 2025). Finally, we include recent frequency-, expert-, and retrieval-based methods FreqMoE, Dynamic-TMoE, and SARAF (Liu, 2025; Zhu et al., 2026; Zhou et al., 2026).

Table 1: Comparison of forecasting performance across eight datasets. Lower values are better. Best results are shown in **bold**, and second-best results are underlined.

Method	Datasets															
	CBW		ECW		NewAPP		Switch		ECL32		ECL57		ECL270		Traffic852	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Ours	**0.101**	**0.115**	**0.062**	**0.122**	**0.029**	0.112	**0.054**	**0.148**	**0.027**	**0.122**	**0.096**	**0.179**	**0.038**	**0.140**	**0.024**	**0.111**
Dynamic-TMoE (2026)	0.208	0.190	0.087	0.144	0.038	0.127	0.064	0.164	0.036	0.145	0.111	0.199	0.039	0.144	0.196	0.176
SARAF (2026)	0.128	0.169	0.107	0.195	0.100	0.234	0.269	0.371	0.110	0.267	0.122	0.235	0.058	0.182	0.289	0.365
MetaEformer (2025)	0.103	0.135	0.068	0.137	0.032	0.121	0.062	0.155	0.029	0.130	0.101	0.193	0.042	0.149	0.128	0.168
FreqMoE (2025)	0.108	0.123	0.081	0.148	**0.029**	**0.108**	0.061	0.167	0.035	0.135	0.115	0.203	0.043	0.152	0.146	0.183
Fremer (2025)	0.163	0.171	0.079	0.141	0.031	0.118	0.069	0.183	0.033	0.133	0.108	0.200	0.043	0.147	0.182	0.170
UniTime (2024)	0.202	0.189	0.105	0.183	0.034	0.118	0.062	0.161	0.037	0.150	0.127	0.214	0.152	0.303	0.117	0.167
PaiFilter (2024)	0.179	0.172	0.079	0.139	0.035	0.121	0.065	0.180	0.035	0.136	0.103	0.193	0.047	0.161	0.289	0.209
FreTS (2023)	0.164	0.156	0.088	0.159	0.044	0.152	0.063	0.174	0.047	0.158	0.108	0.212	0.044	0.150	0.181	0.219
TSMixer (2023)	0.174	0.172	0.099	0.171	0.045	0.154	0.057	0.157	0.036	0.138	0.123	0.212	0.048	0.154	0.156	0.188
TimesNet (2023)	0.155	0.156	0.081	0.143	0.046	0.140	0.066	0.160	0.037	0.146	0.105	0.191	0.048	0.165	0.138	0.160
DLinear (2023)	0.114	0.131	0.123	0.202	0.039	0.133	0.067	0.162	0.062	0.199	0.136	0.226	0.068	0.198	0.098	0.177
SAN (2023)	0.176	0.182	0.178	0.266	25.913	3.696	1.688	0.815	0.038	0.145	0.116	0.209	0.044	0.150	0.185	0.220
Koopa (2023)	0.197	0.192	0.093	0.162	0.058	0.163	0.072	0.176	0.037	0.139	0.130	0.218	0.047	0.156	0.166	0.199
DynEformer (2023)	0.124	0.161	0.067	0.141	0.090	0.207	0.067	0.169	0.063	0.184	0.138	0.239	0.089	0.224	0.144	0.257
VaRDE-L (2023)	0.541	0.455	0.150	0.218	0.556	0.502	0.270	0.369	0.525	0.555	0.379	0.366	0.165	0.285	0.532	0.554
Quatformer (2022)	0.212	0.202	0.192	0.272	0.054	0.158	0.078	0.217	0.168	0.329	0.160	0.263	0.077	0.207	0.106	0.200
FEDformer (2022)	0.249	0.237	0.160	0.257	0.047	0.152	0.126	0.280	0.184	0.346	0.126	0.231	0.060	0.181	0.107	0.198
Non-sta. (2022)	0.203	0.183	0.089	0.175	0.063	0.175	0.068	0.171	0.043	0.161	0.132	0.224	0.047	0.161	0.090	0.133
Autoformer (2021)	0.164	0.193	0.079	0.146	0.111	0.229	0.158	0.270	0.166	0.320	0.141	0.248	0.064	0.188	0.138	0.241

Implementation and evaluation. All experiments are implemented in PyTorch (Paszke et al., 2019) and run on a single NVIDIA A800 80GB PCIe GPU. All methods use the same $48 \rightarrow 24$ setting. We report MSE and MAE (lower is better); all results are averaged over three runs with different random seeds. Data preprocessing and training configurations are detailed in App. A.6.

5.2 Experimental Results and Analysis

Overall results. Table 1 reports the forecasting performance on all eight datasets. At the reported three-decimal precision, SCPEC achieves the best or tied-best result on 15 of the 16 MSE/MAE entries, including 14 sole best results, and ranks second on the remaining one. The strongest baseline, by contrast, varies across entries: seven methods take this role, and the most frequent, MetaEformer, does so on only 6 of the 16. App. C shows example forecasts.

Performance under application-state changes. Relative to our run of MetaEformer (Huang et al., 2025), SCPEC reduces MSE from 0.068 to 0.062 on ECW, from 0.032 to 0.029 on NewAPP, and from 0.062 to 0.054 on Switch, corresponding to relative reductions of 8.82%, 9.38%, and 12.90%, respectively (Fig. 2(c)). The reductions are larger on the two application-event scenarios than on ECW. This pattern is consistent with Sec. 3.2, where application events are associated with weaker historical support and less reliable similarity-based reuse.

The NewAPP MAE exception. The only entry on which SCPEC is not best is the NewAPP MAE: SCPEC ties FreqMoE (Liu, 2025) in MSE (0.029) but trails it in MAE (0.112 vs. 0.108). A spectral diagnostic of the input windows (App. B.1) shows that NewAPP is more spectrally concentrated than ECW: the mean Top-3 spectral concentration ratio, i.e., the share of non-DC spectral energy in the three dominant frequencies, increases from 0.7394 to 0.7766 (5.03%), while the normalized spectral entropy decreases from 0.5626 to 0.5396 (4.09%). The higher concentration of NewAPP persists for $K_s \in \{1, 2, 3, 5, 8\}$ dominant frequencies. Such concentration may be particularly compatible with the frequency-band experts of FreqMoE, which offers a plausible explanation of its small MAE advantage; the analysis shows an association, not a causal link.

Table 2: Ablation study of SCPEC on four representative datasets. Lower values are better, and the best results are highlighted in bold. All results are averaged over three random seeds.

Variant	ECW		NewAPP		Switch		ECL270	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
SCPEC	**0.062**	**0.122**	**0.029**	**0.112**	**0.054**	**0.148**	**0.038**	**0.140**
w/o Pattern-Expert	0.105	0.186	0.047	0.156	0.069	0.182	0.052	0.165
w/o Spectral	0.093	0.162	0.040	0.141	0.065	0.163	0.047	0.159
w/ MPP	0.067	0.132	0.032	0.126	0.059	0.160	0.042	0.147

5.3 Ablation Study

To evaluate the contribution of the key components of SCPEC, we construct three ablated variants. *w/o Pattern-Expert* removes pattern-expert composition, so the model retains no explicit pattern-level knowledge. *w/o Spectral* removes the spectral view from the workload state while retaining the rest of the pipeline. *w/ MPP* replaces the knowledge module of SCPEC, i.e., the learnable bank, state-conditioned routing, and gated fusion, with the Meta-pattern Pooling (MPP) and Echo mechanism of MetaEformer (Huang et al., 2025), including the series decomposition that MPP uses to maintain its pool, while keeping the multi-view workload state, the Transformer encoder, and the horizon-query readout unchanged. It therefore compares history-derived, similarity-based pattern reuse with learnable, state-conditioned composition on the same workload state. We evaluate these variants on ECW, NewAPP, Switch, and ECL270.

As shown in Table 2, the complete SCPEC achieves the best MSE and MAE on all four datasets, and every variant degrades both metrics on every dataset. Removing the pattern experts causes the largest degradation, increasing MSE by 28%–69% across the four datasets. Because this variant removes explicit pattern-level knowledge, the result shows that pattern-expert composition makes a substantial contribution within SCPEC. Removing the spectral view increases MSE by 20%–50%, suggesting that window-level spectral structure provides workload-state information beyond recent temporal dynamics.

Replacing the knowledge module with MPP yields a smaller but consistent degradation of 8%–11% in MSE on every dataset, including NewAPP and Switch. Because both models use the same workload-state representation and backbone, this is a controlled comparison of the two ways of reusing pattern knowledge: learnable, state-conditioned composition consistently outperforms history-derived, similarity-based reuse within this architecture and evaluation, in line with the two challenges characterized in Sec. 3.2.

6 Conclusion and Future Work

In this paper, we revisit reusable knowledge for dynamic multi-node workload forecasting and identify two complementary limitations of history-bound pattern reuse: the *history-bound coverage bottleneck* and the *application-conditioned applicability shift*, which persists in our historical-composition diagnostic. To address them, we move beyond history-derived pattern pools and represent reusable workload knowledge with a learnable pattern-expert bank shared across nodes, from which experts are sparsely selected and composed according to a multi-view workload state built from temporal, spectral, and static-context views. Experiments on eight real-world workload, electricity, and traffic datasets show that SCPEC achieves the best or tied-best result on 15 of the 16 MSE/MAE entries, and its reductions over MetaEformer are larger on the application-event scenarios NewAPP and Switch than on ECW. These results suggest that, in the evaluated settings, reusable forecasting knowledge can be exploited more flexibly through state-conditioned composition than through reuse mechanisms tied to history-derived patterns. Our evaluation covers single-channel workloads and two types of application events on one platform; future work will investigate more efficient expert routing and extend state-conditioned composition toward online adaptation in continually evolving workload environments.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript polishing, LaTeX editing, debugging experimental code, literature and information retrieval, and reference formatting. The core research ideas, model implementation, experimental data, and reported numerical results were produced by the authors. Generative AI was used only as an auxiliary tool to support the authors' implementation and writing process. All AI-assisted text and code were reviewed and verified by the authors. We take full responsibility for the final content of this work, including all text, code, and other artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work uses publicly released system-level workload, electricity, and traffic datasets that, to our knowledge, do not contain personally identifiable information, and we do not foresee ethical concerns beyond those common to time-series forecasting research.

REPRODUCIBILITY STATEMENT

Sec. 5.1 and App. A.6 describe the datasets, chronological splits, preprocessing, metrics, and training setup; App. A complements Sec. 4 with the remaining architectural details, the training objective, and pseudocode (Algorithm 1); and App. B.2 gives the protocol and statistics of the motivating analysis in Sec. 3.2. The implementation code for the main SCPEC experiments is available at `https://anonymous.4open.science/r/test-F170/`.

REFERENCES

Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O. Arik, and Tomas Pfister. TSMixer: An all-MLP architecture for time series forecasting. *Transactions on Machine Learning Research*, 2023.

Weiqi Chen, Wenwei Wang, Bingqing Peng, Qingsong Wen, Tian Zhou, and Liang Sun. Learning to rotate: Quaternion transformer for complicated periodical time series forecasting. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 146–156. ACM, 2022. doi: 10.1145/3534678.3539234.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Seungha Hong, Sukang Chae, Suyeon Kim, Sanghwan Jang, and Hwanjo Yu. Dynamic multi-period experts for online time series forecasting. In *Proceedings of the ACM Web Conference 2026*, pp. 7669–7678, 2026. doi: 10.1145/3774904.3792731.

Shaoyuan Huang, Zheng Wang, Heng Zhang, Xiaofei Wang, Cheng Zhang, and Wenyu Wang. One for all: Unified workload prediction for dynamic multi-tenant edge cloud platforms. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 788–797. ACM, 2023. doi: 10.1145/3580305.3599453.

Shaoyuan Huang, Tiancheng Zhang, Zhongtian Zhang, Xiaofei Wang, Lanjun Wang, and Xin Wang. MetaEformer: Unveiling and leveraging meta-patterns for complex and dynamic systems load forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 991–1002. ACM, 2025. doi: 10.1145/3711896.3737047.

Shaoyuan Huang, Zheng Wang, Heng Zhang, Xiaofei Wang, Cheng Zhang, and Wenyu Wang. DynEformer: A unified framework for robust workload prediction under dynamic environment. *IEEE Transactions on Knowledge and Data Engineering*, 38(7):4239–4253, 2026. doi: 10.1109/TKDE.2026.3689812.

Masoud Kaviani, Parvaneh Asghari, and Maliheh Sabeti. A comprehensive survey and taxonomy of workload forecasting techniques in cloud computing. *Computer Networks*, 286:112454, 2026. doi: 10.1016/j.comnet.2026.112454.

Qianyang Li, Xingjun Zhang, Shaoxun Wang, Tao Peng, and Jia Wei. FAME: Forecastability-aware mixture of experts for heterogeneous time series forecasting. *arXiv preprint arXiv:2606.08896*, 2026.

Yanan Li, Haitao Yuan, Zhe Fu, Xiao Ma, Mengwei Xu, and Shangguang Wang. ELASTIC: Edge workload forecasting based on collaborative cloud-edge deep learning. In *Proceedings of the ACM Web Conference 2023*, pp. 3056–3066, 2023. doi: 10.1145/3543507.3583436.

Yanan Li, Penghong Zhao, Xiao Ma, Haitao Yuan, Zhe Fu, Mengwei Xu, and Shangguang Wang. A collaborative cloud-edge approach for robust edge workload forecasting. *IEEE Transactions on Mobile Computing*, 24(4):2861–2875, 2025a. doi: 10.1109/TMC.2024.3502683.

Yuting Li, Shaoyuan Huang, Tengwen Zhang, Chao Qiu, Mengwei Xu, Cheng Zhang, and Xiaofei Wang. Scout: Tailored collaborative workload forecasting for multi-tenant edge cloud platforms. In *Proceedings of the IEEE International Conference on Communications*, pp. 2346–2351, 2025b. doi: 10.1109/ICC52391.2025.11160848.

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. In *Proceedings of the ACM Web Conference 2024*, pp. 4095–4106. ACM, 2024. doi: 10.1145/3589334.3645434.

Yiwen Liu, Chenyu Zhang, Junjie Song, Siqi Chen, Sun Yin, Zihan Wang, Lingming Zeng, Yuji Cao, and Junming Jiao. MoFE-Time: Mixture of frequency domain experts for time-series forecasting models. *arXiv preprint arXiv:2507.06502*, 2025.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Yong Liu, Chenyu Li, Jianmin Wang, and Mingsheng Long. Koopa: Learning non-stationary time series dynamics with Koopman predictors. In *Advances in Neural Information Processing Systems*, volume 36, 2023a.

Zhiding Liu, Mingyue Cheng, Zhi Li, Zhenya Huang, Qi Liu, Yanhu Xie, and Enhong Chen. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. In *Advances in Neural Information Processing Systems*, volume 36, pp. 14273–14292, 2023b.

Ziqi Liu. FreqMoE: Enhancing time series forecasting through frequency decomposition mixture of experts. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 3430–3438. PMLR, 2025.

Evandro S. Ortigossa, Guy Lutsker, and Eran Segal. MoHETS: Long-term time series forecasting with mixture-of-heterogeneous-experts. *arXiv preprint arXiv:2601.21866*, 2026.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pp. 8024–8035, 2019.

Chenhao Qu, Rodrigo N. Calheiros, and Rajkumar Buyya. Auto-scaling web applications in clouds: A taxonomy and survey. *ACM Computing Surveys*, 51(4):1–33, 2018. doi: 10.1145/3148149.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=B1ckMDqlg`.

Rui Wang, Renhao Xue, Ray Razi, Huan Song, and Hannah R. Marlowe. AME-TS: Anchored mixture-of-experts for time series forecasting. In *Forty-third International Conference on Machine Learning*, 2026.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. TimeXer: Empowering transformers for time series forecasting with exogenous variables. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, volume 34, pp. 22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023.

Xin Wu, Fei Teng, Xingwang Li, Bin Zheng, and Qiang Duan. PRISM: Dynamic primitive-based forecasting for large-scale GPU cluster workloads. In *Proceedings of the 63rd ACM/IEEE Design Automation Conference*, 2026. doi: 10.1145/3770743.3804350.

Eunju Yang and Chan-Hyun Youn. Temporal data pooling with meta-initialization for individual short-term load forecasting. *IEEE Transactions on Smart Grid*, 14(4):3246–3258, 2023. doi: 10.1109/TSG.2022.3225805.

Hengyu Ye, Jiadong Chen, Fuxin Jiang, Xiao He, Tieying Zhang, Jianjun Chen, and Xiaofeng Gao. Fremer: Lightweight and effective frequency transformer for workload forecasting in cloud services. *Proceedings of the VLDB Endowment*, 18(11):3812–3825, 2025. doi: 10.14778/3749646.3749656.

Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Defu Lian, Ning An, Longbing Cao, and Zhendong Niu. Frequency-domain MLPs are more effective learners in time series forecasting. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. FilterNet: Harnessing frequency filters for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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, pp. 11121–11128, 2023. doi: 10.1609/aaai.v37i9.26317.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*, 2023.

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, 2025a. doi: 10.1109/TSC.2025.3536313.

Feiyu Zhao, Weiwei Lin, Shengsheng Lin, Haocheng Zhong, and Keqin Li. TFEGRU: Time-frequency enhanced gated recurrent unit with attention for cloud workload prediction. *IEEE Transactions on Services Computing*, 18(1):467–478, 2025b. doi: 10.1109/TSC.2024.3517324.

Shiqiao Zhou, Holger Schöner, Zipeng Wu, Edouard Fouché, IAG Wilson, and Shuo Wang. Stationarity-aware retrieval-augmented time series forecasting. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 6875–6886, 2026. doi: 10.1145/3770855.3817813.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 27268–27286. PMLR, 2022.

Jiawen Zhu, Shuhan Liu, Di Weng, and Yingcai Wu. Dynamic TMoE: A drift-aware dynamic mixture of experts framework for non-stationary time series forecasting. In *Forty-third International Conference on Machine Learning*, 2026. URL `https://openreview.net/forum?id=JabkBcaoa9`.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. ST-MoE: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.

A Additional Method Details

This section complements Sec. 4 with implementation-level details that the main text omits; equations already given there are not repeated. Algorithm 1 gives the complete forward pass, and the subsections below specify its components.

A.1 Workload-State Construction

Temporal view. The temporal tokens of Sec. 4.2 are

$$\mathbf{T} = \mathcal{E}_{\text{val}}(\mathbf{X}) + \mathcal{E}_{\text{time}}(\mathbf{z}) + \mathbf{E}_{\text{pos}} \in \mathbb{R}^{L\times D}, \tag{10}$$

where $\mathcal{E}_{\text{val}}$, $\mathcal{E}_{\text{time}}$, and $\mathbf{E}_{\text{pos}}$ are value, time-feature, and positional embeddings. The temporal path operates on the complete workload signal rather than on a detrended sequence, so the hidden representation retains workload level, local variation, and temporal shape.

Spectral view. The de-meaned and windowed input $\widetilde{\mathbf{X}}$ of Eq. (2) is

$$\widetilde{X}_{t,c} = w_t\Big(X_{t,c} - \frac{1}{L}\sum_{t'=1}^{L} X_{t',c}\Big), \qquad w_t = \frac{1}{2}\Big(1 - \cos\frac{2\pi(t-1)}{L}\Big), \quad t = 1,\dots,L, \tag{11}$$

i.e., a periodic Hann window. Pool_c averages the real and imaginary parts of the spectrum over channels. In Eq. (3), ϕ_{tf} is a position-wise feed-forward layer and the gate $\mathbf{g}_t$ is a linear map followed by a sigmoid, both applied to $[\mathbf{T}_t; \overline{\mathbf{F}}_t]$.

Context view. The context fusion Φ_{ctx} of Eq. (4) is

$$\mathbf{H}_t^{(0)} = \text{LN}\left(\mathbf{H}_t^{tf} + \text{Dropout}(\phi_s(\mathbf{s}))\right), \tag{12}$$

where ϕ_s linearly projects the static attributes to $\mathbb{R}^D$ and the projection is shared by all tokens.

A.2 Routing, Composition, and Gating

The operations below are applied to each stage ℓ and each local block q of Sec. 4.3; the stage index is dropped where no confusion arises.

Routing. ϕ_{low} is a position-wise linear map from $\mathbb{R}^{D_{\text{exp}}}$ to a routing space of dimension d_r, so that $\mathbf{r}_q \in \mathbb{R}^{3d_r}$ in Eq. (5). The router computes probabilities over all E experts, $\boldsymbol{\pi}_q = \text{Softmax}(\mathbf{a}_q)$, keeps the K largest, and renormalizes them,

$$\alpha_{q,k} = \frac{\pi_{q,e_k}}{\sum_{k'=1}^{K}\pi_{q,e_{k'}}}, \qquad k = 1,\dots,K, \tag{13}$$

which equals the softmax over the selected logits in Eq. (5).

Composition and recovery. Each stage has its own bank $\mathcal{P}^{(\ell)}$. The recovery map in Eq. (6) is a position-wise two-layer network,

$$\phi_{\text{rec}}(\mathbf{v}) = \mathbf{W}_2\,\text{LN}\left(\text{GELU}(\mathbf{W}_1\mathbf{v} + \mathbf{b}_1)\right) + \mathbf{b}_2, \qquad \mathbf{v} = \mathbf{F}_{q,j,:} \in \mathbb{R}^K, \tag{14}$$

so the K weighted pattern values at each position are mapped to $\mathbb{R}^{D_{\text{exp}}}$ without first being summed into a single waveform.

Gated residual. The gate of Eq. (7) is $\boldsymbol{\gamma}_q = \sigma\big(\phi_g([\mathbf{Z}_{\text{exp},q}; \mathbf{M}_q])\big)$, where ϕ_g is a position-wise two-layer network with a GELU activation. The enhanced blocks are concatenated in their original order and joined with the preserved part, giving $\mathbf{H}^{(\ell)} = [\mathbf{Z}^{(\ell)}_{\text{keep}}; \widetilde{\mathbf{Z}}^{(\ell)}_{\text{exp}}]$.

A.3 EMA WARM-START OF THE EXPERT BANK

The bank is randomly initialized. During the first N_{warm} training epochs, the bank is additionally maintained by an exponential moving average of the input workload segments routed to each expert. For block q of a training window, the input segment is the G-step slice of the normalized look-back window aligned with the block, and it is assigned to the block's top-ranked expert e_1. Let $\mathcal{B}_e$ be the set of segments assigned to expert e in the current mini-batch. With decay $\beta \in (0, 1)$, running sums and counts are updated as

$$\mathbf{s}_e \leftarrow \beta\, \mathbf{s}_e + (1-\beta) \sum_{\mathbf{x} \in \mathcal{B}_e} \mathbf{x}, \qquad c_e \leftarrow \beta\, c_e + (1-\beta)\, |\mathcal{B}_e|, \tag{15}$$

and every expert is then overwritten by its running mean $\mathbf{p}_e \leftarrow \mathbf{s}_e / \max(c_e, \epsilon)$. Because this overwrite follows the composition of each block at every warm-up step, the experts track the running means during the warm-up; afterwards, $\mathcal{P}^{(\ell)}$ is updated only by gradients of the training objective.

A.4 TRAINING OBJECTIVE

The forecasting loss is the mean squared error over the horizon,

$$\mathcal{L}_{\text{pred}} = \frac{1}{BHC_o} \sum_{b=1}^{B} \sum_{\tau=1}^{H} \sum_{c=1}^{C_o} \big(\widehat{Y}_{b,\tau,c} - Y_{b,\tau,c}\big)^2, \tag{16}$$

over a mini-batch of B windows with C_o output channels. The routing regularizer combines the load-balancing loss (Fedus et al., 2022) and the router z-loss (Zoph et al., 2022) of sparse Mixture-of-Experts layers. For one block, let $\boldsymbol{\pi}_b$, $\mathbf{a}_b$, and $\mathcal{K}_b$ be the router probabilities, logits, and selected experts of window b; then

$$\mathcal{L}_{\text{lb}} = E \sum_{e=1}^{E} \Big(\frac{1}{B} \sum_{b=1}^{B} \pi_{b,e}\Big)\Big(\frac{1}{BK} \sum_{b=1}^{B} \mathbb{I}[e \in \mathcal{K}_b]\Big), \qquad \mathcal{L}_z = \frac{1}{B} \sum_{b=1}^{B} \Big(\log \sum_{e=1}^{E} \exp a_{b,e}\Big)^2, \tag{17}$$

and $\mathcal{L}_{\text{aux}}$ averages $\mathcal{L}_{\text{lb}} + \lambda_z \mathcal{L}_z$ over blocks and stages. The complete objective is $\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda_{\text{aux}} \mathcal{L}_{\text{aux}}$.

A.5 COMPUTATIONAL COMPLEXITY

Per look-back window, the temporal embedding costs $\mathcal{O}(L(C + d_z)D)$ and the optional static view $\mathcal{O}(d_s D + LD)$. The spectral view costs $\mathcal{O}(CL \log L)$ for the FFT, $\mathcal{O}(K_f D + D^2)$ for ϕ_f, and $\mathcal{O}(LD^2)$ for the token-wise gated fusion of Eq. (3). Each of the M Transformer stages costs $\mathcal{O}(L^2 D + LD^2 + LDd_{\text{ff}})$, where d_{ff} is the feed-forward width. In each stage, the pattern-expert branch costs $\mathcal{O}(LD_{\text{exp}}d_r + (L/G)Ed_r)$ for routing, $\mathcal{O}(LK)$ for gathering and weighting the $K \ll E$ selected patterns, $\mathcal{O}(L(K + D_{\text{exp}})d_h)$ for the recovery map ϕ_{rec} with hidden width d_h, and $\mathcal{O}(LD_{\text{exp}}^2)$ for the gate ϕ_g. The horizon-query readout costs $\mathcal{O}(HLD + (H + L)D^2 + HDd_{\text{ff}})$. For fixed E, G, and hidden widths, the pattern-expert branch is therefore linear in L and adds no super-linear dependence on the look-back length beyond the $\mathcal{O}(L^2 D)$ attention term of the Transformer backbone; moreover, E enters its cost only through the router scores, since composition and recovery touch only the K selected patterns.

Measured training cost. To complement this analysis, we measure the training cost of SCPEC and MetaEformer (Huang et al., 2025) (our run of its official code) on ECW under the same $48 \to 24$ setting and otherwise identical configurations, on a single NVIDIA A800 80GB PCIe GPU. As shown in Table 3, SCPEC uses 28.0% less GPU memory and completes the run 27.3% faster.

A.6 IMPLEMENTATION DETAILS

Data preprocessing. All splits are chronological and follow the data pipeline of the MetaEformer code base (Huang et al., 2025). For CBW and ECW, the first 80% of each series is used for training and the last 20% for testing, input windows are extracted with a stride of 12, and inputs are min–max scaled with statistics fitted on the training portion. The electricity datasets use a 7:1:2 and Traffic852

Table 3: Training cost on ECW ($48 \rightarrow 24$) on a single NVIDIA A800 80GB PCIe GPU with otherwise identical configurations. GPU memory is as reported by `nvidia-smi`; relative changes are with respect to MetaEformer.

Method	GPU memory (MiB)	Run time
MetaEformer	6605	29 min 4.9 s
SCPEC	**4753** (-28.0%)	**21 min 8 s** (-27.3%)

an $8{:}1{:}1$ train/validation/test split, with z-score normalization fitted on the training portion. NewAPP and Switch are used only as target application-event scenarios for zero-shot evaluation of models trained on ECW. MSE and MAE are computed on the original scale for the cloud-workload datasets and on the normalized scale for the electricity and traffic datasets.

Training. SCPEC is trained with Adam and a one-cycle learning-rate schedule with maximum learning rate 10^{-4}.

B Additional Analysis for Experimental Results

B.1 Detailed Spectral Diagnostics

This section provides the protocol, figure, and complete statistics for the spectral diagnostic summarized in Sec. 5.2. The purpose of this diagnostic is to examine whether the local performance exception observed on NewAPP is associated with distinguishable input-side spectral characteristics.

Diagnostic protocol. We follow the same forecasting setting used in the main experiments. Each diagnostic sample consists of a 48-step input window associated with a valid 24-step prediction horizon, and consecutive windows are extracted with a stride of 12. For ECW, diagnostic windows are constructed from the held-out test portion, whereas NewAPP and Switch use their complete target sequences following the zero-shot evaluation protocol. This yields $4{,}782$, 594, and 12 diagnostic windows for ECW, NewAPP, and Switch, respectively.

For each input window $\mathbf{x}_i$, we first remove its temporal mean and apply a Hann window $\mathbf{w}$ before computing the real-valued Fourier transform. The power of the k-th non-DC frequency component is

$$P_{i,k} = \left|\mathrm{rFFT}\left(\mathbf{w} \odot (\mathbf{x}_i - \bar{\mathbf{x}}_i)\right)_k\right|^2, \tag{18}$$

which is normalized as

$$p_{i,k} = \frac{P_{i,k}}{\sum_{j=1}^{F} P_{i,j} + \epsilon}, \tag{19}$$

where F denotes the number of non-DC frequency bins and ϵ is a small constant for numerical stability, so that $\sum_{k=1}^{F} p_{i,k} \approx 1$.

We quantify the concentration of spectral energy using the Top-K_s spectral concentration ratio

$$\mathrm{SCR}_{K_s}(\mathbf{x}_i) = \sum_{k \in \mathrm{TopK}(\mathbf{p}_i, K_s)} p_{i,k}, \tag{20}$$

and normalized spectral entropy

$$H_{\mathrm{spec}}(\mathbf{x}_i) = -\frac{\sum_{k=1}^{F} p_{i,k} \log(p_{i,k} + \epsilon)}{\log F}. \tag{21}$$

where K_s is the number of dominant frequencies (distinct from the number K of active experts in Sec. 4.3). A larger SCR_{K_s} indicates that a greater fraction of spectral energy is captured by a small number of dominant frequencies, whereas a smaller H_{spec} indicates a less dispersed spectral distribution.

Algorithm 1: Forward pass of SCPEC

Input: Look-back window $\mathbf{X} \in \mathbb{R}^{L \times C}$ ($C = 1$ in our experiments), temporal covariates $\mathbf{z}$, optional static context $\mathbf{s}$, learnable pattern-expert banks $\mathcal{P}^{(\ell)} = \{\mathbf{p}_e^{(\ell)} \in \mathbb{R}^G\}_{e=1}^{E}$ for $\ell = 1, \dots, M$, learnable horizon queries $\mathbf{Q} \in \mathbb{R}^{H \times D}$, number of activated experts K, application ratio η, and block length G dividing L

Output: H-step forecast $\hat{\mathbf{Y}} \in \mathbb{R}^{H \times C}$

/* Multi-view workload state */

$\mathbf{T} \leftarrow E_{\text{val}}(\mathbf{X}) + E_{\text{time}}(\mathbf{z}) + E_{\text{pos}}$; // temporal view

$\widetilde{\mathbf{X}} \leftarrow \mathbf{w} \odot (\mathbf{X} - \bar{\mathbf{X}})$; // de-mean, Hann window

$\mathbf{F}_{K_f} \leftarrow \Pi_{K_f}\big(\mathrm{rFFT}(\widetilde{\mathbf{X}})\big)$;

$\mathbf{f} \leftarrow \phi_f\big([\mathrm{Pool}_c \mathrm{Re}(\mathbf{F}_{K_f}); \mathrm{Pool}_c \mathrm{Im}(\mathbf{F}_{K_f})]\big)$; // spectral view

for $t \leftarrow 1$ **to** L **do**

  $\mathbf{g}_t \leftarrow \sigma(\mathbf{W}_f[\mathbf{T}_t; \mathbf{f}] + \mathbf{b}_f)$;

  $\mathbf{H}_t^{tf} \leftarrow \mathbf{T}_t + \mathbf{g}_t \odot \phi_{tf}([\mathbf{T}_t; \mathbf{f}])$; // token-wise gate

if *the static view is used* **then**

  $\mathbf{H}^{(0)} \leftarrow \Phi_{\text{ctx}}\big(\mathbf{H}^{tf}, \phi_s(\mathbf{s})\big)$; // static-context view

else

  $\mathbf{H}^{(0)} \leftarrow \mathbf{H}^{tf}$;

/* State-conditioned sparse pattern-expert composition */

$D_{\text{exp}} \leftarrow \lfloor \eta D \rfloor$;

for $\ell \leftarrow 1$ **to** M **do**

  $\mathbf{Z}^{(\ell)} \leftarrow \mathcal{T}^{(\ell)}\big(\mathbf{H}^{(\ell-1)}\big)$; // Transformer encoding

  $(\mathbf{Z}_{\text{keep}}, \mathbf{Z}_{\text{exp}}) \leftarrow \mathrm{Split}_\eta\big(\mathbf{Z}^{(\ell)}\big)$; // $\mathbf{Z}_{\text{exp}} \in \mathbb{R}^{L \times D_{\text{exp}}}$

  foreach $q \in \{1, \dots, L/G\}$ **do**

    $\mathbf{Z}_{\text{exp},q} \leftarrow \mathbf{Z}_{\text{exp}}[(q-1)G + 1 : qG]$; // local workload block

    $\mathbf{A}_q \leftarrow \phi_{\text{low}}(\mathbf{Z}_{\text{exp},q})$;

    $\mathbf{r}_q \leftarrow [\mathrm{Mean}(\mathbf{A}_q); \mathrm{Std}(\mathbf{A}_q); \mathrm{Max}(\mathbf{A}_q)]$; // over the G positions

    $\mathbf{a}_q \leftarrow \mathbf{W}_r \mathbf{r}_q + \mathbf{b}_r$;

    $(e_1, \dots, e_K) \leftarrow \mathrm{TopK}(\mathbf{a}_q, K)$; // in descending order of score

    $\boldsymbol{\alpha}_q \leftarrow \mathrm{Softmax}(a_{q,e_1}, \dots, a_{q,e_K})$;

    for $j \leftarrow 1$ **to** G **do**

      for $k \leftarrow 1$ **to** K **do**

        $F_{q,j,k} \leftarrow \alpha_{q,k}\,[\mathbf{p}_{e_k}^{(\ell)}]_j$;

      $\mathbf{M}_{q,j} \leftarrow \phi_{\text{rec}}(\mathbf{F}_{q,j,:})$; // $\mathbb{R}^K \to \mathbb{R}^{D_{\text{exp}}}$

    $\boldsymbol{\gamma}_q \leftarrow \sigma\big(\phi_g([\mathbf{Z}_{\text{exp},q}; \mathbf{M}_q])\big)$;

    $\widetilde{\mathbf{Z}}_{\text{exp},q} \leftarrow \mathbf{Z}_{\text{exp},q} + \boldsymbol{\gamma}_q \odot (\mathbf{M}_q - \mathbf{Z}_{\text{exp},q})$; // gated residual

    if *training, within the first* N_{warm} *epochs* **then**

      EMA-update the statistics of e_1 with $\mathbf{X}[(q-1)G + 1 : qG]$; reset $\mathcal{P}^{(\ell)}$ to the running means; // App. A.3

  $\mathbf{H}^{(\ell)} \leftarrow [\mathbf{Z}_{\text{keep}}; \widetilde{\mathbf{Z}}_{\text{exp}}]$; // restore blocks

/* Horizon-query readout */

$\bar{\mathbf{H}} \leftarrow \mathrm{LN}_{\text{enc}}(\mathbf{H}^{(M)})$;

$\mathbf{R}_1 \leftarrow \mathrm{LN}_1\big(\mathrm{MHA}(\mathbf{Q}, \bar{\mathbf{H}}, \bar{\mathbf{H}})\big)$;

$\widetilde{\mathbf{R}} \leftarrow \mathrm{LN}_2\big(\mathbf{R}_1 + \mathrm{FFN}(\mathbf{R}_1)\big)$;

$\hat{\mathbf{Y}} \leftarrow \mathbf{W}_o \widetilde{\mathbf{R}} + \mathbf{b}_o$;

return $\hat{\mathbf{Y}}$;

Complete spectral statistics. Table 4 reports the complete spectral-concentration statistics underlying Fig. 4. NewAPP consistently exhibits the largest mean spectral concentration for all examined values of K_s, while also having the lowest normalized spectral entropy. Relative to ECW, NewAPP

Table 4: Detailed spectral statistics for the ECW-derived scenarios. Higher $\overline{\mathrm{SCR}}_{K_s}$ and lower $\overline{H}_{\mathrm{spec}}$ indicate stronger spectral concentration. N_w denotes the number of diagnostic input windows.

Scenario	N_w	**Spectral Concentration Ratio**					$\overline{H}_{\mathrm{spec}}$
		SCR_1	SCR_2	SCR_3	SCR_5	SCR_8	
ECW	4782	0.4210	0.6085	0.7394	0.8821	0.9454	0.5626
NewAPP	594	**0.4262**	**0.6427**	**0.7766**	**0.8984**	**0.9537**	**0.5396**
Switch	12	0.3976	0.6325	0.7618	0.8787	0.9410	0.5671

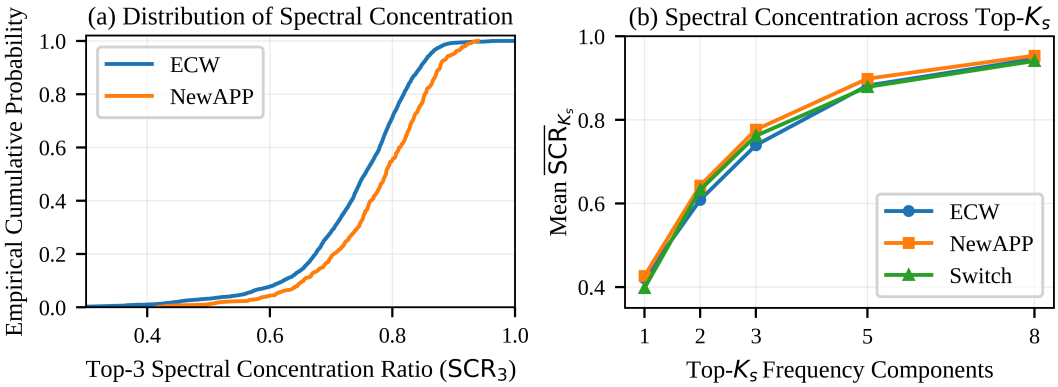


Figure 4: Spectral characteristics under application-state changes. (a) Empirical cumulative distributions of the Top-3 spectral concentration ratio (SCR_3) on ECW and NewAPP. NewAPP exhibits an overall shift toward stronger spectral concentration. (b) Mean spectral concentration under different choices of K_s. The stronger concentration of NewAPP remains consistent across $K_s \in \{1, 2, 3, 5, 8\}$.

increases $\overline{\mathrm{SCR}}_3$ from 0.7394 to 0.7766 (5.03% higher), while its normalized spectral entropy decreases from 0.5626 to 0.5396 (4.09% lower). These two statistics provide complementary evidence: the former measures the energy captured by dominant frequencies, whereas the latter characterizes the dispersion of the complete normalized spectrum.

Robustness to the choice of K_s. The main text uses $K_s = 3$ as a compact summary of dominant-frequency concentration. To verify that the observation is not specific to this particular choice, we additionally evaluate $K_s \in \{1, 2, 3, 5, 8\}$. As shown in Table 4 and Fig. 4(b), NewAPP maintains a larger $\overline{\mathrm{SCR}}_{K_s}$ than ECW across all examined values of K_s. In particular, the difference becomes more apparent for $K_s \in \{2, 3\}$ and remains present as more frequency components are included. Therefore, the stronger spectral concentration observed on NewAPP is not an artifact of selecting $K_s = 3$.

Distribution-level comparison. Beyond comparing dataset-level means, Fig. 4(a) reports the empirical cumulative distribution function (ECDF) of SCR_3. For a scenario $\mathcal{D}$ containing $N_{\mathcal{D}}$ diagnostic windows, we compute

$$\widehat{F}_{\mathcal{D}}(s) = \frac{1}{N_{\mathcal{D}}} \sum_{i=1}^{N_{\mathcal{D}}} \mathbb{I}\left[\mathrm{SCR}_3(\mathbf{x}_i) \le s\right]. \tag{22}$$

The ECDF of NewAPP is shifted toward larger spectral-concentration values relative to ECW. This distribution-level behavior indicates that the increase in mean SCR_3 is broadly reflected across NewAPP input windows rather than being determined solely by a small number of highly concentrated samples.

Interpretation with frequency-specialized forecasting. FreqMoE dynamically decomposes time-series signals into frequency bands and processes different bands using specialized experts (Liu, 2025). The stronger spectral concentration observed on NewAPP therefore provides an input-side characteristic that may be particularly compatible with such frequency-specialized

modeling. This observation offers a plausible explanation for the small NewAPP MAE advantage of FreqMoE reported in Table 1, while SCPEC remains tied for the best MSE at the reported precision. We emphasize that this analysis establishes an association between spectral characteristics and model inductive bias rather than a causal relationship. The diagnostic is computed solely from the input workloads and does not rely on post-hoc manipulation of model predictions.

Remark on Switch. The Switch scenario contains only 12 valid diagnostic windows under the same 48-step input, 24-step prediction, and stride-12 protocol. Consequently, its spectral statistics are included in Table 4 for completeness, but we treat them as descriptive rather than drawing strong distributional conclusions from this small sample. The ECDF comparison in Fig. 4(a) therefore focuses on ECW and NewAPP, for which substantially larger numbers of diagnostic windows are available.

B.2 Protocol and Statistics of the Motivating Analysis

Protocol. Each ECW series is split chronologically into a training part (first 80%) and a test part (last 20%). Windows of $L = 48$ input steps followed by $H = 24$ future steps are extracted with stride 12. Every training window of every node enters the reference set $\mathcal{R}$; the queries are the windows of the ECW test parts and all windows of NewAPP and Switch, giving $4{,}782$, 594, and 12 queries, respectively. For matching, each window is reduced to its seasonal component, i.e., the window minus its edge-padded moving average with kernel size 24, and Sim is the Pearson correlation of the last 16 steps of these components. For an ECW query, pool windows from the same node are excluded so that a node cannot match its own past; this rule can only lower ECW coverage, so the coverage gaps in Sec. 3.2 are conservative. Futures are compared on raw values by the Pearson correlation Corr. Formally, the two rates of Sec. 3.2 are

$$\mathrm{Cov}(\rho) = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbb{I}\big[\mathrm{Sim}\big(q, r^\star(q)\big) \geq \rho\big],$$

$$\mathrm{PTA} = \widehat{\Pr}_{q \in \mathcal{Q}}\Big[\mathrm{Corr}\big(\mathbf{y}_q, \mathbf{y}_{r^\star(q)}\big) < 0.5 \;\Big|\; \mathrm{Sim}\big(q, r^\star(q)\big) \geq 0.9\Big].$$

Statistics. Table 5 reports the rates of Sec. 3.2 with 95% node-level bootstrap intervals ($10{,}000$ resamples of nodes), which account for the overlap of windows within a node. Intervals condition on the fixed ECW training reference set and treat nodes as independent resampling units within each dataset; with only 11 NewAPP nodes, they are approximate. The ECW–NewAPP differences remain clearly separated: the PTA ratio of NewAPP is $6.32\times$ that of ECW, with a bootstrap interval of $[4.52, 9.07]$. Switch, with 12 windows from 3 nodes, is reported descriptively. The raw-workload mean/median behind the ECDFs in Fig. 2(b1) are $1.174/0.803$ for ECW, $0.385/0.253$ for NewAPP, and $0.287/0.187$ for Switch.

Composition of historical windows. Pattern pools can compose several patterns instead of reusing a single match (Huang et al., 2025). As a diagnostic of history-based composition, each query is also approximated by a greedy non-negative combination of K windows among its 64 most similar reference windows: the weights are fitted by non-negative least squares on the query's seasonal tail, and the same weights combine the corresponding futures. The diagnostic transfers futures directly and does not reproduce how a forecaster such as MetaEformer decodes composed patterns. With $K = 4$, composition largely restores input coverage: $\mathrm{Cov}(0.95)$ reaches 97.8% on ECW and 97.0% on NewAPP, with overlapping intervals (Table 5). It also lowers the NewAPP PTA ratio from 17.77% to 11.78%, but this remains $3.83\times$ the ECW ratio of 3.08% (bootstrap interval $[2.67, 5.59]$), and 9 of 11 covered Switch queries are PTA cases; with $K = 2$ the ratios are 2.93% and 9.29%. In this diagnostic, historical-window composition substantially improves input coverage but does not eliminate the elevated future-transfer ambiguity on NewAPP.

C Visualization of Forecasting Results

This section provides additional visualizations of SCPEC forecasting results, complementing the aggregate quantitative evaluation in the main paper with a trajectory-level view. For each dataset, we

Table 5: Rates of the motivating analysis with 95% node-level bootstrap intervals (10,000 resamples of nodes). Rows marked "composed" use a greedy non-negative combination of $K = 4$ windows among the 64 most similar reference windows. Switch is reported as counts.

	ECW	NewAPP	Switch
Cov(0.90)	95.2% [94.3, 96.0]	91.9% [88.0, 95.3]	7/12
Cov(0.95)	85.3% [83.8, 86.8]	66.3% [61.4, 71.0]	3/12
PTA ratio	2.81% [2.11, 3.58]	17.77% [14.16, 22.44]	4/7
Cov(0.95), composed	97.8% [97.2, 98.3]	97.0% [95.5, 98.5]	10/12
PTA ratio, composed	3.08% [2.28, 3.96]	11.78% [9.09, 14.81]	9/11

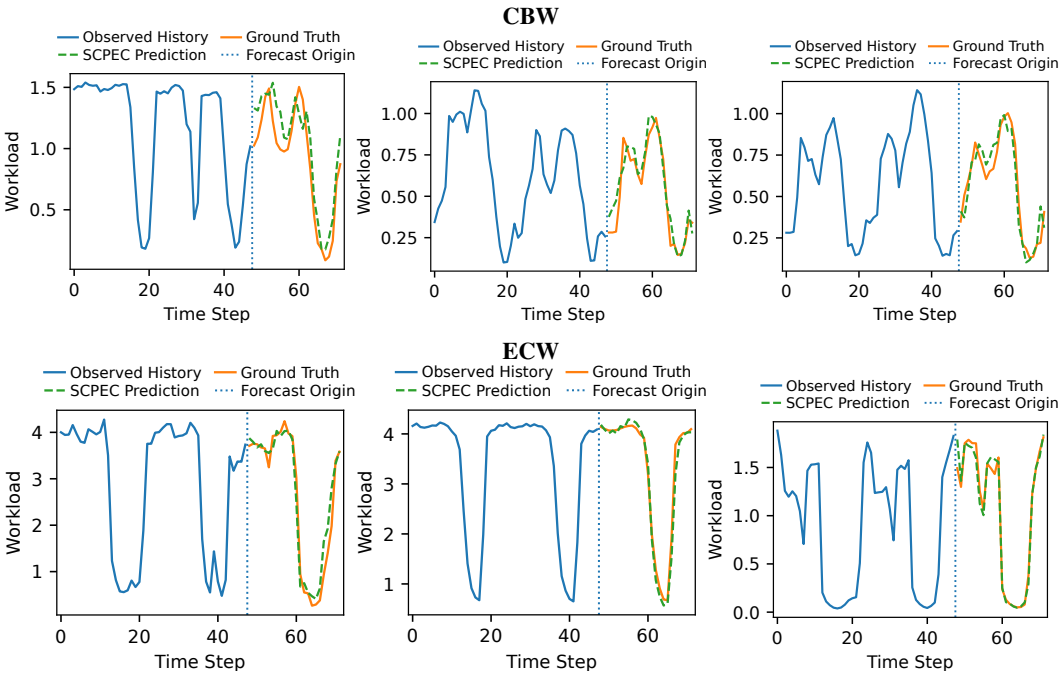


Figure 5: Example SCPEC forecasts on three test windows each of CBW (top) and ECW (bottom).

select three forecasting cases from the first test batch. Each case shows the observed workload history before the forecast origin together with the ground-truth future and the SCPEC prediction over the forecasting horizon, where the vertical dotted line marks the forecast origin. Visualizing multiple cases allows us to inspect the model under different local workload states, fluctuation magnitudes, and temporal evolution patterns rather than relying on a single forecasting case.

CBW and ECW. Fig. 5 presents forecasting results on CBW and ECW. Each row corresponds to one dataset, while the three panels show different cases. These examples cover different local cloud-workload dynamics, including changes in workload level, peak and valley locations, and transitions around the forecast origin. The visualizations provide a direct view of how SCPEC follows the dominant future trend and major turning points across these cases.

NewAPP and Switch. Fig. 6 presents forecasting results on the two dynamic application scenarios. Again, three cases from the first test batch are shown for each dataset. Compared with the regular ECW setting, NewAPP and Switch reflect workload changes associated with application events and therefore provide a more challenging qualitative evaluation. Following the protocol in the main paper, both scenarios are evaluated using the model trained on ECW without target-scenario retraining. The displayed trajectories illustrate SCPEC's forecasting behavior under different post-event workload states and local temporal changes.

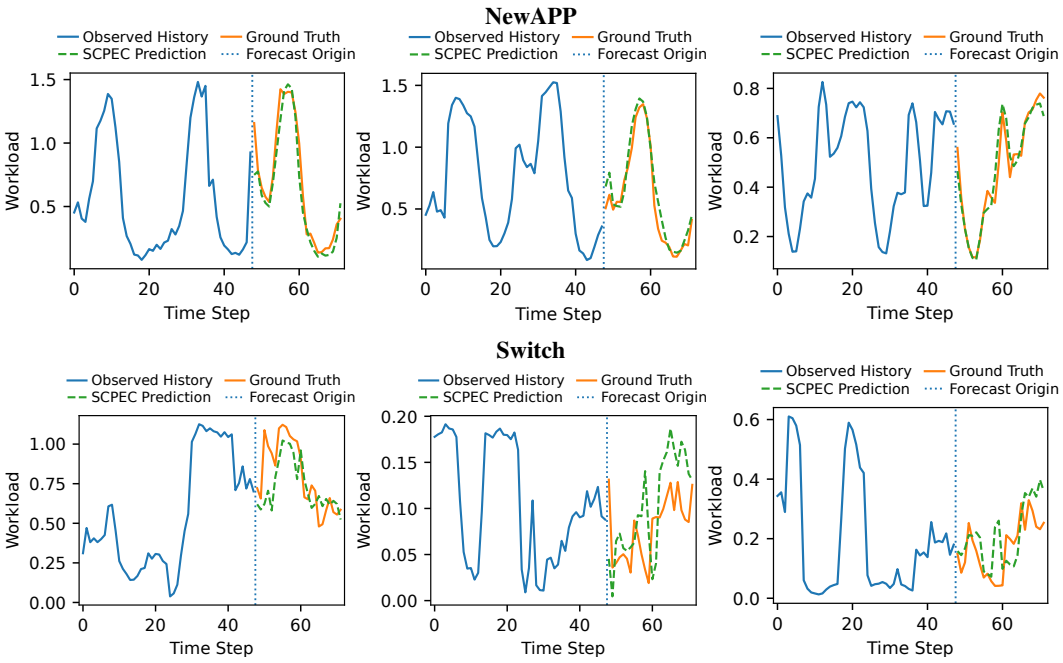


Figure 6: Example SCPEC forecasts on three test windows each of NewAPP (top) and Switch (bottom), using the model trained on ECW without retraining.

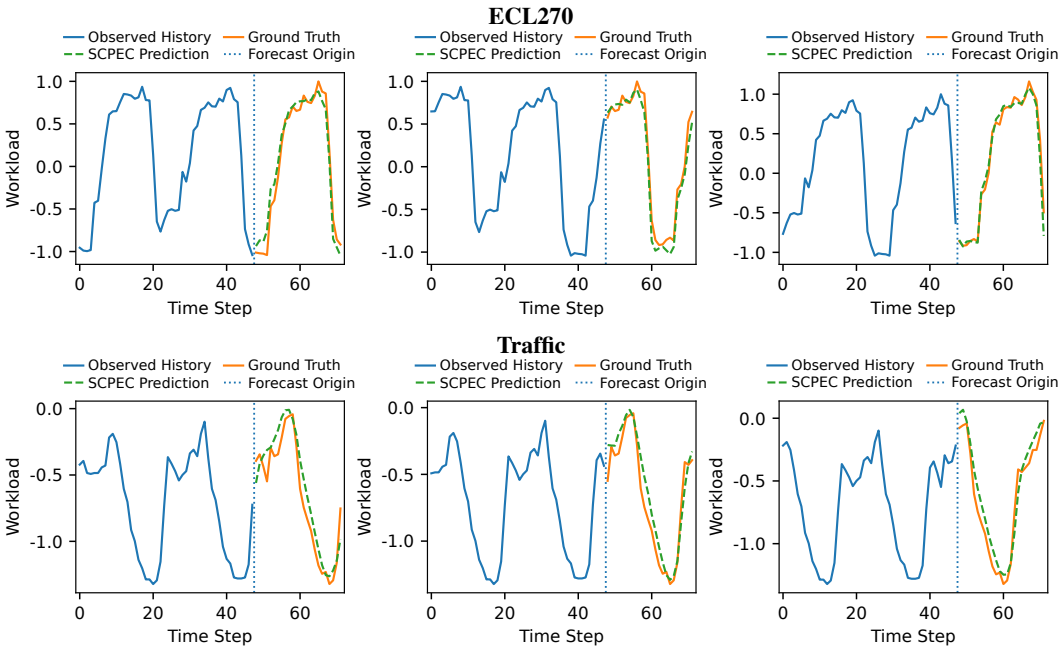


Figure 7: Example SCPEC forecasts on three test windows each of ECL270 (top) and Traffic852 (bottom).

ECL270 and Traffic852. Fig. 7 presents corresponding visualizations on ECL270 and Traffic852. These datasets come from application domains different from the cloud-workload setting and exhibit distinct periodic structures, amplitude variations, and local temporal patterns. For each dataset, three forecasting cases from the first test batch are shown. Together with the MSE/MAE results reported in the main paper, these examples provide a complementary trajectory-level view of SCPEC's forecasting behavior on electricity-load and traffic time series.