SCALING MULTIVARIATE TIME SERIES FORECASTING WITH CONTEXT-MODULATED PROTOTYPE ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multivariate time series forecasting is fundamental to a wide range of real-world applications, from traffic and energy systems to large-scale retail platforms, where hundreds to tens of thousands of interacting variables must be forecast jointly. Full pairwise attention captures rich cross-variate interactions but incurs quadratic complexity in the number of variates, while efficient alternatives reduce this cost by restricting or compressing cross-variate communication, potentially sacrificing flexible, input-dependent interactions. Real-world forecasting further requires incorporating known covariates, such as calendar events and promotions, and characterizing predictive uncertainty. We propose SLIM, a unified Transformer architecture built on context-modulated prototype attention (CoPA). CoPA compresses cross-variate interactions into a small set of latent prototypes formed from the history, and lets known covariates modulate only the queries that retrieve from them, so that the same shared representation is read differently under different forecast contexts. This yields cost linear in the number of variates, adds a covariate path that is independent of the prototype budget, and supports quantile forecasting within a single backbone. Extensive experiments across short-term, long-term, high-dimensional, and context forecasting benchmarks demonstrate state-of-the-art accuracy with substantially lower memory and computational costs.

1 INTRODUCTION

Multivariate time series forecasting plays a critical role in numerous real-world applications, including traffic management (Zhang et al., 2024a), climate science (Wu et al., 2023b), and financial analytics (Zhang et al., 2024b). A central challenge in these settings is to model dependencies among many interacting variates. Transformer-based architectures (Liu et al., 2024; Zhang & Yan, 2023) provide a natural mechanism for this purpose by allowing each variate to dynamically aggregate information from others. However, full pairwise attention incurs quadratic complexity in the number of variates, making it increasingly expensive as dimensionality grows.

Existing efficient approaches reduce this cost by restricting or compressing cross-variate communication, for example through channel-independent processing (Zeng et al., 2023), MLP-based mixing (Chen et al., 2023), centralized aggregation (Han et al., 2024), or channel clustering (Fang et al., 2026; Qiu et al., 2025; Chen et al., 2024). These designs can substantially improve computational efficiency, but they also constrain how information is exchanged across variates. This creates a fundamental trade-off: full attention provides flexible, data-dependent interaction modeling but scales poorly as the number of variates grows, whereas efficient alternatives often achieve lower cost by simplifying the interaction structure. An important question is therefore whether we can retain flexible cross-variate interactions without paying the quadratic cost of full attention.

Beyond cross-variate modeling, real-world forecasting must also account for known covariates (Cao et al., 2026) and predictive uncertainty (Ansari et al., 2025). Known covariates, such as calendar events, promotions, and weather forecasts, are available over both the look-back window and the forecast horizon, and can change both the temporal dynamics of individual series and the dependencies among them, so the information relevant to a forecast depends on its context. Many downstream decisions also require predictive distributions rather than point estimates. Existing architectures typically handle cross-variate interaction, covariate conditioning, and uncertainty estimation through separate mechanisms, leading to fragmented designs and additional computational overhead.

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.

To address these challenges, we propose SLIM, a unified Transformer architecture built on context-modulated prototype attention (CoPA). Following the latent-token bottlenecks used in efficient attention (Lee et al., 2019; Jaegle et al., 2021), CoPA compresses cross-variate information into $r \ll N$ latent prototypes and lets each variate retrieve from this shared representation. Its key design choice is to separate two operations that a single attention map would entangle: how the shared cross-variate representation is formed, and how each variate reads from it. CoPA forms the prototypes from historical observations alone, and lets known covariates, over both the look-back window and the forecast horizon, modulate only the retrieval queries of the final layer. This conditioning path is linear in N, independent of the prototype budget r, and evaluated once per forward pass. The resulting representation is decoded into multiple quantiles, so that cross-variate modeling, covariate conditioning, and quantile forecasting share a single backbone.

Our contributions are three-fold. First, we propose CoPA, a prototype-based cross-variate interaction mechanism whose cost $O(Nrd)$ is linear in the number of variates for a fixed budget r, while retrieval remains input dependent. Second, we specify where known covariates should enter such a mechanism: prototypes are formed from the history alone and covariates modulate only how variates retrieve from them. We show that this makes the retrieval pattern context dependent and gives the model the form of a conditional latent-factor model. Third, we evaluate SLIM on short-term, long-term, high-dimensional, and covariate-aware benchmarks, together with memory and runtime measurements and ablations that isolate the placement of the covariate signal.

2 Related Work

Early deep forecasters based on recurrence and convolution (Liu et al., 2022) mainly model temporal dynamics. Transformers such as Informer (Zhou et al., 2021), Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022) and PatchTST (Nie et al., 2023) improve temporal efficiency through sparse attention, decomposition, frequency-domain modeling or patching, but leave the cost of cross-variate interaction unaddressed. Channel-dependent methods such as iTransformer (Liu et al., 2024) and Crossformer (Zhang & Yan, 2023) attend across variates and achieve strong accuracy at quadratic cost, whereas channel-independent methods (Nie et al., 2023; Zeng et al., 2023) avoid this cost by processing each variate separately and discarding cross-variate information. Recent efficient alternatives seek a middle ground: MLP mixing and centralized aggregation (Chen et al., 2023; Han et al., 2024) compress the exchange, while clustering-based methods (Chen et al., 2024; Qiu et al., 2025; Fang et al., 2026) organize variates into learned groups. These designs reduce cost, but may constrain fine-grained or input-dependent interactions.

A standard route to linear cost is to attend through a small set of latent tokens. Set Transformer attends through learned inducing points (Lee et al., 2019), Perceiver attends from a fixed latent array to the input (Jaegle et al., 2021), and Linformer projects keys and values to a fixed length (Wang et al., 2020). In forecasting, Crossformer routes cross-dimension messages through a few router tokens, and U-Cast uses hierarchical latent queries for high-dimensional series (Ni et al., 2025). CoPA builds on this idea, but forms its prototypes as input-dependent soft aggregations of the variates and specifies which inputs may shape them.

Covariate-aware models such as TFT (Lim et al., 2021), TiDE (Das et al., 2023) and NBEATSx (Olivares et al., 2023) encode known covariates alongside the target series, and TimeXer (Wang et al., 2024) injects exogenous variables through cross-attention. These methods determine how covariates are encoded, but not where they should act inside a cross-variate interaction mechanism. CoPA applies them as a FiLM-style feature-wise affine modulation (Perez et al., 2018) of the retrieval queries only, leaving the shared prototypes untouched. For uncertainty, DeepAR (Salinas et al., 2020) and Chronos (Ansari et al., 2025) produce full predictive distributions, whereas SLIM predicts a fixed set of quantiles from the same backbone with the pinball loss.

3 Methodology

Given a look-back window of L time steps over N variates, $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_L\} \in \mathbb{R}^{L \times N}$, the goal is to predict the next S steps $\mathbf{Y} = \{\mathbf{x}_{L+1}, \ldots, \mathbf{x}_{L+S}\} \in \mathbb{R}^{S \times N}$. The N variates are forecast jointly, allowing the model to capture dependencies across them. Many real-world forecasting problems additionally provide *known covariates*, such as calendar information, public holidays, weather

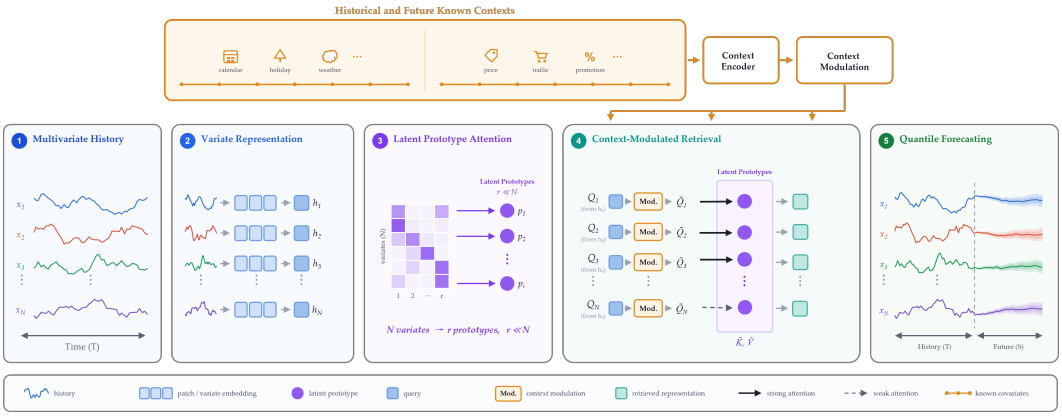

Figure 1: Overview of SLIM: latent prototypes are formed from the history alone, and known covariates modulate only the retrieval queries of the final layer.

forecasts, or promotion schedules, whose values are available over both the look-back window and the forecast horizon. We stack all such channels into a single tensor $\mathbf{C} \in \mathbb{R}^{(L+S)\times N\times M}$, where M is the number of covariate channels; a channel that is common to all variates, such as a calendar field, is copied along the variate dimension. The model treats every channel identically, and $\mathbf{C}$ is the only input available over the forecast horizon.

Figure 1 illustrates a single forward pass of SLIM. Each variate is first encoded from its historical observations into a variate-level representation. These representations are then softly aggregated into $r \ll N$ latent prototypes, replacing quadratic pairwise cross-variate attention with interaction through a compact prototype space. The prototypes in every layer are constructed solely from history-derived representations, while known covariates modulate how each variate queries the prototype space in the final layer. Each variate thus retrieves a context-dependent mixture of shared prototypes, which is decoded into forecasts at multiple quantile levels.

3.1 Variate Representation

We first construct a variate-level representation from the historical observations of each variate. The historical sequence is segmented into patches of length p with stride $s \le p$, producing $P = \lfloor (L-p)/s \rfloor + 1$ patches per variate. The patches overlap when $s < p$. To incorporate temporal information, each patch is augmented with a normalized relative-time encoding, forming $\tilde{\mathbf{X}}_n \in \mathbb{R}^{P\times 2p}$. Each patch is mapped into a d-dimensional embedding through a residual projection block:

$$\mathbf{Z}_n = \mathrm{ResidualBlock}(\tilde{\mathbf{X}}_n), \qquad \mathbf{Z}_n \in \mathbb{R}^{P\times d}. \tag{1}$$

The patch embeddings are then concatenated and projected into a single variate-level representation:

$$\mathbf{H}_n = \mathrm{Linear}\left([\mathbf{Z}_{n,1}; \ldots; \mathbf{Z}_{n,P}]\right), \qquad \mathbf{H} \in \mathbb{R}^{N\times d}. \tag{2}$$

Here, $\mathbf{H}_n \in \mathbb{R}^d$ denotes the representation of variate n, and $\mathbf{H}$ depends on the historical observations $\mathbf{X}$ alone.

3.2 Latent Prototype Attention

The prototypes are constructed only from the historical representations. For a fixed historical observation, this keeps the shared prototype representation unchanged when the known covariates vary. The covariates are introduced later to modulate how each variate retrieves information from this shared prototype space. For clarity we describe a single layer and a single attention head. We obtain queries, keys, and values through linear projections:

$$\mathbf{Q} = \mathbf{H}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{H}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{H}\mathbf{W}_V, \qquad \mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{N\times d}, \tag{3}$$

where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d}$ are learnable projection matrices. To capture cross-variate interactions efficiently, we compress keys and values into r latent prototypes through a learned assignment

matrix. A shared linear projection first maps each variate embedding to assignment logits, which are then converted into soft assignments by a Gumbel-perturbed softmax (Jang et al., 2017):

$$\tilde{\mathbf{A}} = \mathrm{Linear}(\mathbf{H}) \in \mathbb{R}^{N\times r}, \qquad \mathbf{A} = \mathrm{softmax}_n\left((\tilde{\mathbf{A}} + \alpha\mathbf{G})/\tau\right) \in \mathbb{R}^{N\times r}. \tag{4}$$

Here $\tilde{\mathbf{A}}_{n,j}$ scores variate n against prototype j, and $\mathbf{G}_{n,j} = -\log(-\log U_{n,j})$ with $U_{n,j} \overset{\text{i.i.d.}}{\sim} \mathrm{Uniform}(0,1)$ is Gumbel noise added during training only. The softmax is taken over the variate index n, so that $\sum_{n=1}^{N} \mathbf{A}_{n,j} = 1$ for $j = 1, \ldots, r$. Each prototype is thus a weighted aggregation over all N variates rather than a hard partition of them. The learnable temperature τ controls the concentration of the assignments, while α controls the scale of the injected noise. Applying $\mathbf{A}$ to the keys and values gives the r prototypes:

$$\bar{\mathbf{K}} = \mathbf{A}^\top\mathbf{K}, \qquad \bar{\mathbf{V}} = \mathbf{A}^\top\mathbf{V}, \qquad \bar{\mathbf{K}}, \bar{\mathbf{V}} \in \mathbb{R}^{r\times d}. \tag{5}$$

Since $\mathbf{A}$ is normalized over the variate dimension, each prototype is a convex combination of the corresponding keys or values, which prevents its scale from growing with N. The prototype bottleneck reduces the cost of cross-variate interaction from $O(N^2 d)$ to $O(Nrd)$ when $r \ll N$. The Gumbel perturbation keeps the assignment differentiable while introducing stochasticity into prototype formation during training; at inference $\mathbf{A}$ is deterministic.

3.3 Context-Modulated Retrieval

Known covariates describe the conditions under which a forecast is made, over the look-back window and over the forecast horizon, where they carry information that the historical target series alone does not determine. We use these covariates to modulate how each variate retrieves information from the shared prototype space, while keeping $\mathbf{A}$, $\bar{\mathbf{K}}$, and $\bar{\mathbf{V}}$ determined solely by the historical representations. For each variate n, a learned basis compresses the time axis of its covariates into $k \ll L + S$ coefficients:

$$\mathbf{c}_n = \mathrm{Linear}_k\left(\mathbf{C}_{:,n,:}\right), \qquad \mathbf{c}_n \in \mathbb{R}^{Mk}. \tag{6}$$

This temporal compression preserves information about when known events occur over the look-back and forecast horizons. The coefficients are projected to a d-dimensional context representation and then mapped to a scale and shift for the corresponding query:

$$\mathbf{u}_n = \mathrm{Linear}\left(\mathbf{c}_n\right), \qquad [\boldsymbol{\gamma}_n; \boldsymbol{\beta}_n] = \mathrm{Linear}(\mathbf{u}_n), \qquad \tilde{\mathbf{Q}}_n = (1 + \boldsymbol{\gamma}_n) \odot \mathbf{Q}_n + \boldsymbol{\beta}_n, \tag{7}$$

where $\boldsymbol{\gamma}_n, \boldsymbol{\beta}_n \in \mathbb{R}^d$. The final projection is initialized to zero, so that training starts from $\tilde{\mathbf{Q}} = \mathbf{Q}$. The modulated queries then attend to the prototype keys and values:

$$\mathbf{H}' = \mathrm{softmax}\left(\frac{\tilde{\mathbf{Q}}\bar{\mathbf{K}}^\top}{\sqrt{d}}\right)\bar{\mathbf{V}}, \qquad \mathbf{H}' \in \mathbb{R}^{N\times d}, \tag{8}$$

where the softmax is taken row-wise, over the r prototypes. Each variate receives its own $\boldsymbol{\gamma}_n$ and $\boldsymbol{\beta}_n$, allowing its retrieval pattern to vary with its known covariates. A covariate common to all variates is copied along the variate dimension and therefore influences every query. The context path contains only per-variate projections and introduces no additional pairwise cross-variate interaction.

Stacking. SLIM stacks ℓ such layers, each followed by a residual connection, layer normalization and a feed-forward block, and the output of layer l is the input $\mathbf{H}^{(l+1)}$ of layer $l+1$. Only the final layer uses the modulated queries $\tilde{\mathbf{Q}}$; the first $\ell - 1$ layers use the unmodulated queries $\mathbf{Q}$. Every layer input $\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(\ell)}$ is therefore a function of $\mathbf{X}$ alone, and so is every assignment $\mathbf{A}^{(l)}$ and every prototype $\bar{\mathbf{K}}^{(l)}, \bar{\mathbf{V}}^{(l)}$ in the network. The covariates enter exactly once, when the final layer retrieves from its prototypes, so the context path is evaluated once per forward pass regardless of depth. Without covariates, $\tilde{\mathbf{Q}} = \mathbf{Q}$ in every layer. With h attention heads, $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ are split into h blocks of width d/h that share one assignment $\mathbf{A}$ per layer.

3.4 Quantile Head and Optimization

The output of the final layer, $\mathbf{H}'^{(\ell)}$, is projected onto the forecasting horizon by a residual quantile prediction head:

$$\hat{\mathbf{Y}}^{\mathcal{Q}} = \mathrm{ResidualBlock}\big(\mathbf{H}'^{(\ell)}\big) \in \mathbb{R}^{N\times S\times|\mathcal{Q}|}. \tag{9}$$

The model is optimized using the pinball loss:

$$\mathcal{L}_{\text{quantile}} = \frac{1}{NS|\mathcal{Q}|} \sum_{n=1}^{N} \sum_{t=1}^{S} \sum_{q \in \mathcal{Q}} \left[q \max\left(y_{n,t} - \hat{y}^q_{n,t}, 0\right) + (1-q) \max\left(\hat{y}^q_{n,t} - y_{n,t}, 0\right) \right]. \quad (10)$$

Here, $\mathcal{Q} = \{0.1, 0.2, \ldots, 0.9\}$ denotes the set of nine predicted quantile levels, $\hat{y}^q_{n,t}$ is the forecast for variate n at horizon step t and quantile level q, and $y_{n,t}$ is the corresponding ground-truth value. At inference the nine outputs are sorted along the quantile axis, which removes quantile crossing without affecting the median. The predicted quantiles characterize predictive uncertainty, and the median forecast at $q = 0.5$ is used as the point prediction.

3.5 Theoretical Insights

Proposition 1 *Fix the historical observation* $\mathbf{X}$*. A prototype retrieval mechanism whose queries depend only on the layer input* $\mathbf{H}^{(\ell)}$ *produces a retrieval matrix* $\mathbf{R} = \mathrm{softmax}(\mathbf{Q}\bar{\mathbf{K}}^\top/\sqrt{d})$ *that is independent of the known covariates* $\mathbf{C}$*. In contrast, the context-modulated queries* $\tilde{\mathbf{Q}}$ *produce different retrieval matrices for the same* $\mathbf{X}$ *whenever the covariates change the relative attention logits over the prototypes.*

The proof is given in Appendix A. Proposition 1 shows that conditioning the retrieval queries on known covariates allows the same historical representation to induce different retrieval patterns under different forecast contexts. SLIM therefore keeps prototype formation dependent only on the history, and uses the covariates to modulate the queries that retrieve from the shared prototype space.

Proposition 2 *The context-modulated prototype retrieval in SLIM admits a conditional latent-factor representation. The output of the final layer for variate* n *can be written as*

$$\mathbf{H}'^{(\ell)}_n = \sum_{j=1}^{r} \mathbf{R}_{n,j}(\mathbf{X}, \mathbf{C})\, \bar{\mathbf{V}}^{(\ell)}_j(\mathbf{X}), \quad (11)$$

where $\bar{\mathbf{V}}^{(\ell)}_j(\mathbf{X})$ *is a history-dependent shared prototype and* $\mathbf{R}_{n,j}(\mathbf{X}, \mathbf{C})$ *is its context-dependent retrieval weight. This is the factorized form of a conditional latent-factor model, in which the shared factors are determined by the historical observations and their mixing coefficients depend on the forecast context.*

The proof is given in Appendix A. Proposition 2 gives a structural interpretation of the design: the prototype values carry shared representations derived from the history, while the context-dependent retrieval weights determine how these representations are combined for each variate under different forecast contexts.

4 Experiments

Our experiments address five questions. We first examine whether pooling the variates into r prototypes preserves forecasting accuracy when full pairwise attention is still computationally feasible (Sections 4.1 and 4.2), and whether this behavior persists as the number of variates grows to the thousands (Section 4.3). We then compare SLIM with forecasting models designed to use known covariates (Section 4.4), and measure the additional memory and runtime cost of the covariate path (Section 4.5). Section 4.6 studies the prototype budget r. Finally, we test whether the gains arise specifically from conditioning the retrieval queries, rather than from conditioning the prototypes (Section 4.7). Appendix G provides representative forecasts behind the aggregate results.

4.1 Short-term Forecasting Results

As shown in Table 1, SLIM achieves the best accuracy on all four PEMS subsets under both MSE and MAE. Against the strongest baseline on each subset, it reduces the average MSE by 20.4% on PEMS03 (0.082 vs. 0.103, DUET), 2.2% on PEMS04 (0.090 vs. 0.092, SCINet), 29.1% on PEMS07 (0.061 vs. 0.086, TimeXer and SOFTS), and 15.0% on PEMS08 (0.096 vs. 0.113, DUET). These results indicate that routing cross-variate interaction through a compact set of latent prototypes retains effective communication among sensors without modeling every pairwise interaction.

Table 1: Short-term multivariate time series forecasting results. The input sequence length is $L = 96$ for all models; results are averaged over four forecasting horizons $S \in \{12, 24, 48, 96\}$. Best and second-best results are highlighted in **bold** and underlined, respectively. Per-horizon results are reported in Appendix H.

Dataset	SLIM Ours		DUET 2025		CycleNet 2024		TimeXer 2024		SOFTS 2024		iTransformer 2024		PatchTST 2023		Crossformer 2023		TimesNet 2023		DLinear 2023		SCINet 2022	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	**0.082**	**0.182**	0.103	0.212	0.112	0.214	0.118	0.225	0.104	0.210	0.113	0.222	0.180	0.291	0.169	0.282	0.146	0.248	0.278	0.375	0.114	0.224
PEMS04	**0.090**	**0.186**	0.106	0.217	0.119	0.232	0.105	0.209	0.102	0.208	0.111	0.221	0.195	0.307	0.209	0.314	0.129	0.241	0.295	0.388	0.092	0.202
PEMS07	**0.061**	**0.153**	0.090	0.196	0.113	0.214	0.086	0.182	0.086	0.184	0.101	0.204	0.211	0.303	0.235	0.315	0.124	0.226	0.329	0.395	0.119	0.217
PEMS08	**0.096**	**0.176**	0.113	0.217	0.150	0.246	0.175	0.251	0.138	0.220	0.150	0.226	0.280	0.321	0.268	0.306	0.193	0.271	0.379	0.416	0.159	0.244

4.2 Long-term Forecasting Results

Table 2: Long-term multivariate time series forecasting results. The look-back length is fixed at $L = 96$ and results are averaged over four forecasting horizons $S \in \{96, 192, 336, 720\}$. Best and second-best results are highlighted in **bold** and underlined, respectively. Per-horizon results are reported in Appendix H.

Dataset	SLIM Ours		DUET 2025		CycleNet 2024		TimeXer 2024		SOFTS 2024		iTransformer 2024		PatchTST 2023		Crossformer 2023		TimesNet 2023		DLinear 2023		SCINet 2022	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.430**	**0.428**	0.443	0.436	0.457	0.441	0.437	0.437	0.449	0.443	0.454	0.448	0.469	0.455	0.529	0.522	0.458	0.450	0.456	0.452	0.747	0.647
ETTh2	**0.354**	**0.385**	0.372	0.395	0.388	0.409	0.368	0.396	0.373	0.400	0.383	0.407	0.387	0.407	0.942	0.683	0.414	0.427	0.559	0.515	0.954	0.723
ETTm1	**0.366**	**0.391**	0.390	0.393	0.386	0.395	0.382	0.397	0.393	0.402	0.407	0.410	0.387	0.400	0.513	0.495	0.400	0.406	0.403	0.407	0.485	0.481
ETTm2	**0.254**	**0.314**	0.280	0.324	0.272	0.315	0.289	0.331	0.287	0.330	0.288	0.332	0.281	0.326	0.757	0.611	0.291	0.333	0.350	0.401	0.571	0.537
Weather	**0.229**	**0.262**	0.251	0.273	0.243	0.271	0.240	0.271	0.255	0.278	0.258	0.278	0.259	0.273	0.259	0.315	0.259	0.286	0.265	0.317	0.292	0.363
Solar	**0.205**	**0.222**	0.237	0.233	0.210	0.261	0.237	0.302	0.229	0.256	0.233	0.262	0.236	0.266	0.641	0.639	0.301	0.319	0.330	0.401	0.282	0.375
Electricity	**0.163**	0.263	0.172	**0.259**	0.168	**0.259**	0.171	0.270	0.174	0.264	0.178	0.270	0.205	0.290	0.244	0.334	0.193	0.295	0.212	0.300	0.571	0.537

Table 2 reports results averaged over four horizons on seven widely adopted benchmarks. SLIM achieves the best MSE on all seven datasets and the best MAE on six, ranking second on Electricity MAE, 0.004 behind DUET and CycleNet. Against the strongest baseline on each dataset, the MSE reduction ranges from 1.6% on ETTh1 (TimeXer) to 6.6% on ETTm2 (CycleNet), and SLIM outperforms the channel-dependent iTransformer (Liu et al., 2024) and SOFTS (Han et al., 2024) on every dataset under both metrics. These results suggest that compact prototype-based interaction is an effective alternative to full pairwise cross-variate modeling over long horizons, allowing each variate to selectively retrieve shared information.

4.3 High-dimensional Forecasting Results

Table 3: High-dimensional multivariate time series forecasting results, where "–" indicates that the model exhausted device memory. Solar here is the 5,162-variate Time-HD collection, distinct from the 137-variate Solar of Table 2. Best and second-best results are highlighted in **bold** and underlined, respectively.

Dataset	SLIM (ours)		U-CAST (2025)		DUET (2025)		PAttn (2024)		iTransformer (2024)		TSMixer (2023)		PatchTST (2023)		DLinear (2023)		TimesNet (2023)	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Air	**0.416**	**0.414**	0.446	0.430	0.452	0.444	0.449	0.432	0.447	0.431	0.447	0.438	0.448	0.432	0.449	0.446	0.457	0.438
Measles	**0.010**	**0.038**	**0.010**	0.042	0.015	0.064	0.011	0.048	**0.010**	0.048	0.569	0.547	0.013	0.058	0.128	0.252	0.018	0.060
SP500	**0.454**	**0.279**	0.555	0.328	0.568	0.335	0.516	0.309	0.511	0.306	2.674	1.120	0.523	0.313	0.630	0.367	0.611	0.343
Atec	**0.280**	**0.273**	0.287	0.280	0.330	0.339	0.299	0.275	0.345	0.319	0.398	0.387	0.298	0.298	0.318	0.314	0.493	0.429
Neurolib	**1.681**	**0.325**	1.750	0.350	2.519	0.451	2.458	0.445	1.718	0.347	2.240	0.532	2.395	0.438	1.793	0.381	2.475	0.458
Meter	0.968	**0.545**	0.943	0.551	1.308	0.731	**0.941**	0.552	0.949	0.556	0.987	0.564	1.254	0.706	0.944	0.549	1.034	0.586
SIRS	**0.002**	**0.036**	0.007	0.052	0.095	0.236	0.025	0.109	0.028	0.113	0.016	0.078	0.033	0.129	0.058	0.168	0.162	0.327
M5	**3.387**	**0.831**	3.501	0.849	3.768	0.880	3.650	0.867	3.549	0.853	6.863	1.623	3.655	0.872	3.688	0.870	4.490	0.919
Temp	**0.259**	**0.380**	0.262	0.383	0.435	0.511	0.278	0.395	0.265	0.386	0.266	0.389	0.279	0.396	0.272	0.391	0.287	0.408
Wind	**1.104**	**0.688**	**1.104**	0.692	1.227	0.746	1.256	0.758	1.116	0.699	1.346	0.742	1.254	0.757	1.128	0.697	1.161	0.708
Solar	**0.148**	**0.203**	0.172	0.246	–	–	0.604	0.582	0.343	0.427	0.155	0.216	0.416	0.469	0.174	0.255	0.157	0.224
Mobility	**0.307**	**0.311**	0.315	0.317	0.439	0.410	0.337	0.336	0.312	0.314	1.165	0.787	0.344	0.341	0.344	0.359	0.410	0.388
Traffic-CA	**0.057**	**0.131**	0.061	**0.131**	–	–	0.491	0.554	0.271	0.391	0.082	0.186	0.295	0.417	0.063	0.141	0.101	0.205
Traffic-GBA	**0.056**	**0.125**	0.059	0.126	0.650	0.638	0.059	0.132	0.063	0.141	0.074	0.168	0.060	0.139	0.062	0.137	0.098	0.204
Wiki-20k	**10.273**	**0.283**	**10.273**	0.302	10.278	0.304	10.290	0.306	10.933	0.405	10.446	0.332	10.291	0.305	10.740	0.394	10.586	0.325

The high-dimensional benchmark in Table 3 stresses scalability: the variate count ranges from N=1,105 on Air to N=7,491 on Traffic-CA and N=20,000 on Wiki-20k. SLIM achieves the best MAE on all 15 datasets and the best MSE on 14, with Meter as the only exception. On Solar and Traffic-CA, where the clustering-based DUET exhausts device memory, SLIM runs at full

dimensionality. Compared with U-CAST, the strongest baseline overall, SLIM reduces MSE by 18.2% on SP500, 13.9% on Solar, 71.4% on SIRS and 6.7% on Air, and matches it on Wind and Wiki-20k. By replacing full pairwise interaction with retrieval through $r \ll N$ latent prototypes, SLIM retains input-dependent cross-variate communication at substantially lower interaction cost.

4.4 Covariate-aware Forecasting Results

Table 4: Covariate-aware forecasting results. All methods receive the same known covariates over the look-back window and the forecast horizon. Best and second-best results are highlighted in **bold** and underlined, respectively.

Dataset	SLIM (ours)		TiDE		TSMixerX		TFT		PatchTST + CovMLP		iTransformer + CovMLP		DLinear + CovMLP	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
M5-cov	**0.3054**	**0.2404**	0.3131	0.2416	0.3566	0.2619	0.3085	0.2452	0.3115	0.2456	0.3122	0.2446	0.3183	0.2451
Favorita	**0.4210**	**0.1730**	0.4266	0.1788	0.5196	0.2044	0.4320	0.1784	0.4324	0.1763	0.4343	0.1790	0.4460	0.1820
FreshRetailNet-Daily	**0.2544**	**0.3015**	0.2570	0.3029	0.3667	0.3649	0.2647	0.3074	0.2778	0.3191	0.2735	0.3152	0.2815	0.3191

Table 4 compares SLIM with forecasting models that also consume known covariates, under identical covariate inputs. SLIM achieves the lowest MSE and MAE on all three datasets, reducing MSE by 1.0% on M5-cov (0.3054 vs. 0.3085, TFT), 1.3% on Favorita (0.4210 vs. 0.4266, TiDE) and 1.0% on FreshRetailNet-Daily (0.2544 vs. 0.2570, TiDE). The margins over the strongest covariate-aware models are small, so absolute accuracy alone does not show where the covariate signal should enter; Section 4.7 isolates this question with a controlled comparison.

4.5 Efficiency and Memory

This section measures how prototype compression scales with the number of variates, and separately the marginal cost of the covariate path. We compare with three representative interaction mechanisms: iTransformer, which performs full attention across variates; DUET, which restricts cross-variate interaction through learned clustering; and SOFTS, which exchanges information through a single series core. To isolate cross-variate scaling, all methods use the same tensor shape and are measured sequentially on the same physical GPU within each dataset.

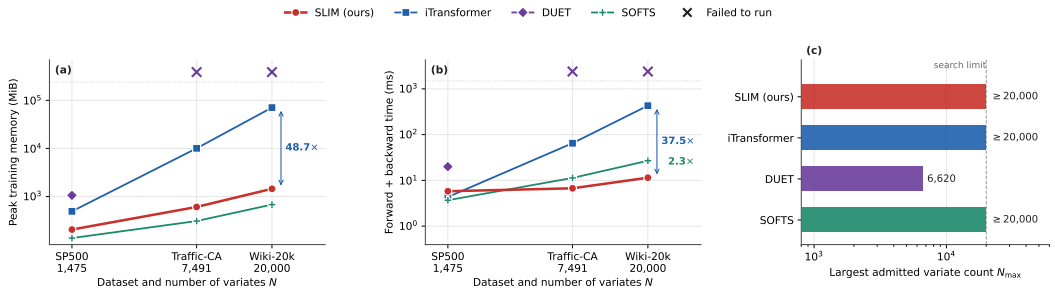

Figure 2: Common-shape efficiency on real SP500 (N=1,475), Traffic-CA (N=7,491) and Wiki-20k (N=20,000) windows: (a) peak allocated training memory, (b) forward/backward step time and (c) the largest admitted variate count $N_{\max}$, obtained by bisection over nested Wiki-20k subsets up to 20,000. All methods use L=96, S=7, batch size one, d=256, d_{ff}=1,024, eight heads, two layers and BF16 training; SOFTS uses d_{core}=256. Points show the median over five independently initialized repeats; error bars give the interquartile range of the step time. Ratios at N=20,000 are relative to SLIM. Crosses mark failed runs: DUET exceeds a cuBLAS size limit on Traffic-CA and runs out of memory on Wiki-20k.

Full attention. The cost of full variate attention grows quadratically with N, while that of SLIM grows roughly linearly. From SP500 to Wiki-20k, a 13.6× increase in N, the memory of iTransformer grows by 143× and its step time by 99×, whereas SLIM grows by 7.0× and 2.0×. At Wiki-20k, SLIM uses 1,448 MiB and 11.45 ms per step against 70,583 MiB and 429.05 ms, reductions of 48.7× and 37.5×. At the smallest scale, iTransformer is slightly faster (4.33 versus 5.79 ms). At this size both methods are dominated by fixed per-step overhead rather than by cross-variate interaction:

the step time of SLIM changes by less than 1 ms from N=1,475 to N=7,491, and its pooling and retrieval run as several small kernels, whereas full attention runs as one fused kernel. Once N reaches a few thousand, the quadratic term dominates and SLIM is faster by an order of magnitude.

Clustering. DUET is the most expensive method at SP500 (1,060 MiB and 19.96 ms, 5.1× and 3.4× the cost of SLIM) and fails beyond $N_{\max}$=6,620 variates, so it cannot run on Traffic-CA or Wiki-20k. SLIM, iTransformer and SOFTS all run at the 20,000-variate limit of the search.

Series-core fusion. SOFTS is the most memory-efficient of the four methods, using about half the memory of SLIM at every scale (680 MiB at Wiki-20k), because it compresses all variates into a single core. It is also the fastest at SP500, but its step time grows faster with N, and SLIM is 1.7× faster at Traffic-CA and 2.3× faster at Wiki-20k. The additional memory of SLIM buys input-dependent interaction through r prototypes rather than one global core, and SLIM is more accurate than SOFTS on every short- and long-term benchmark, reducing MSE by 23.1% and 7.8% on average (Tables 1 and 2). Its absolute memory remains small, below 1.5 GB at 20,000 variates.

Together, these results show that prototype compression removes the quadratic growth of full variate attention and the capacity limit of clustering-based interaction, and gives the best runtime scaling among the four methods at a modest memory cost relative to the most compressed alternative. These measurements use two layers for every method, whereas the accuracy experiments use three for SLIM; since every layer has the same $O(Nrd)$ interaction cost, depth changes the constants but not the scaling.

4.6 Sensitive Analysis

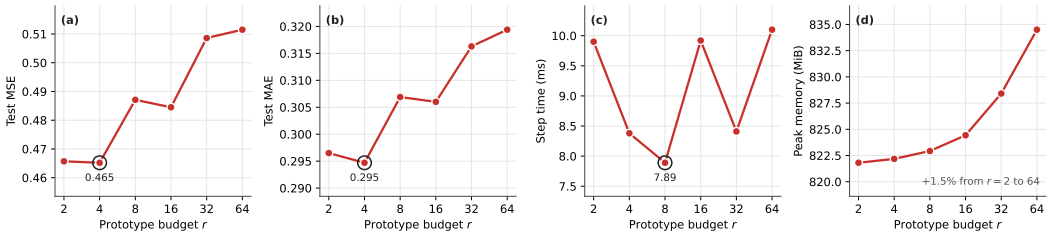

Figure 3: Sensitivity to the prototype budget r on SP500 with L=336: (a) test MSE, (b) test MAE, (c) step time and (d) peak allocated memory. Step time and memory report the median over five repeats; circles mark the best value in each panel.

The prototype budget r is the only hyperparameter controlling the degree of compression. We vary r on SP500 while keeping all other settings fixed, and report forecasting accuracy together with training cost in Figure 3. Performance is strongest at small prototype budgets and gradually deteriorates as r increases. In particular, r=2 and r=4 perform nearly identically: r=4 achieves the lowest MSE, improving over r=2 by only 0.0006, whereas increasing the budget to r=32 or r=64 raises MSE by approximately 9–10%. Rather than suggesting that four prototypes are universally optimal, these results indicate that a small prototype set is already sufficient to capture the dominant shared cross-variate structure in SP500, while additional capacity provides diminishing utility. The validation retrieval statistics support this interpretation: the effective utilization of the available prototype budget decreases from 99.9% at r=2 to 75.8% at r=64, suggesting increasing redundancy as the budget grows. Importantly, this additional capacity incurs only modest memory overhead: peak memory increases monotonically by just 1.5% from r=2 to r=64. Step time is non-monotonic, reflecting differences in CUDA kernel selection across tensor shapes rather than a systematic increase with the prototype budget.

4.7 Ablation Study

Every baseline in Table 4 already consumes the same covariates, so a comparison of absolute accuracy cannot say where the covariate signal belongs. We therefore hold the backbone, the prototype budget and the training protocol fixed, and vary only what the covariates cover and where they are applied.

Three contrasts follow, each removing one link in the claim. The first removes the covariate path entirely, which asks whether it contributes anything beyond the backbone. The second supplies the covariates over the look-back window only, withholding the segment that describes the forecast horizon. The third keeps the full covariates but applies them, at the same final layer, to the prototypes instead of the queries, which tests the placement itself.

We run the comparison on Favorita, the most event-rich of the covariate datasets, selected on validation data before any test evaluation. Because the margins being separated are small, we report, alongside the test errors, 95% bootstrap intervals for the paired difference in test MSE, computed once by resampling series and once by resampling forecast horizons. A margin produced by a few unusual series, or by one part of the horizon, would show up as an interval that includes zero.

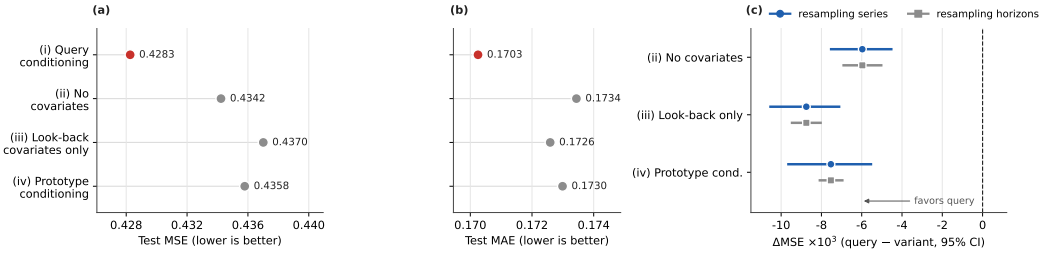

Figure 4: Where the covariate signal enters, on Favorita. All variants share the backbone, prototype budget and training protocol, and differ only in what the covariates cover and where they act. (a, b) Test MSE and MAE of the four variants. (c) Paired difference in test MSE, query conditioning minus each variant, with 95% bootstrap intervals obtained by resampling series and by resampling forecast horizons; intervals below zero favor query conditioning.

As shown in Figure 4(a, b), query conditioning attains the lowest MSE and MAE of the four variants. Removing the covariates raises MSE by 1.37%, so the path contributes beyond the backbone. Withholding the horizon segment raises it by 2.00%, the largest increase; since this variant still receives the covariates over the look-back window, the gain is attributable to information about the forecast horizon rather than to the extra input channels. Applying the covariates to the prototypes instead of the queries raises MSE by 1.73%. Figure 4(c) shows that every interval lies entirely below zero under both resampling schemes, so none of these margins is driven by a few series or by one part of the horizon.

The third contrast is deliberately generous to the alternative. Prototype conditioning modulates the pooled keys, the pooled values and the pooling logits of the final layer, three sites rather than the single query site, so it acts on more of the computation while receiving exactly the same covariates. It is nonetheless worse on both metrics, which supports the proposed placement: the covariates should shape how each variate reads from the shared prototypes, not the prototypes themselves.

5 Conclusion

We present SLIM, a scalable architecture for multivariate time series forecasting that replaces quadratic cross-variate attention with interaction through a compact set of latent prototypes. The key design separates history-based prototype formation from context-dependent retrieval: known covariates modulate how each variate accesses the shared prototype space without changing the prototypes themselves. This yields $O(Nrd)$ cross-variate interaction, while the conditioning path scales linearly with the number of variates and independently of the prototype budget. Across short-term, long-term, and high-dimensional benchmarks, SLIM achieves strong forecasting accuracy while substantially reducing the memory and computational cost of full attention, and scales to as many as 20,000 variates. Covariate-aware experiments and controlled ablations further support the proposed placement of contextual information at retrieval. Future work could explore more diverse prototype representations and finer-grained temporal conditioning to capture richer context-dependent interactions while preserving the favorable scaling properties of the framework.

REFERENCES

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, et al. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025.

Fanpu Cao, Lu Dai, Jindong Han, and Hui Xiong. Enhancing multivariate time series forecasting with global temporal retrieval. *arXiv preprint arXiv:2602.10847*, 2026.

Jialin Chen, Jan Eric Lenssen, Aosong Feng, Weihua Hu, Matthias Fey, Leandros Tassiulas, Jure Leskovec, and Rex Ying. From similarity to superiority: Channel clustering for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023.

Yuchen Fang, Shiyu Wang, Yuxuan Liang, Zhou Ye, Yang Xiang, Yan Zhao, and Kai Zheng. Efficient high-dimensional time series forecasting with transformers: A channel reordering perspective. In *Proceedings of the ACM Web Conference 2026*, pp. 7223–7234, 2026.

Li Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. SOFTS: Efficient multivariate time series forecasting with series-core fusion. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Oriol Vinyals, Andrew Zisserman, and João Carreira. Perceiver: General perception with iterative attention. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 4651–4664, 2021.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, 2017.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International conference on learning representations*, 2021.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant input. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 3744–3753, 2019.

Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International journal of forecasting*, 37(4): 1748–1764, 2021.

Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. Cyclenet: Enhancing time series forecasting through modeling periodic patterns. *Advances in Neural Information Processing Systems*, 37:106315–106345, 2024.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. SCINet: Time series modeling and forecasting with sample convolution and interaction. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024.

Juntong Ni, Shiyu Wang, Zewen Liu, Xiaoming Shi, Xinyue Zhong, Zhou Ye, and Wei Jin. U-CAST: Learning hierarchical structures for high-dimensional time series forecasting. *arXiv preprint arXiv:2507.15119*, 2025.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

Kin G Olivares, Cristian Challu, Grzegorz Marcjasz, Rafał Weron, and Artur Dubrawski. Neural basis expansion analysis with exogenous variables: Forecasting electricity prices with nbeatsx. *International Journal of Forecasting*, 39(2):884–900, 2023.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, 2018.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. DUET: Dual clustering enhanced multivariate time series forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3): 1181–1191, 2020.

Mingtian Tan, Mike A Merrill, Vinayak Gupta, Tim Althoff, and Thomas Hartvigsen. Are language models actually useful for time series forecasting? *Advances in Neural Information Processing Systems*, 37:60162–60191, 2024.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

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. *Advances in Neural Information Processing Systems*, 37:469–498, 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, 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*, 2023a.

Haixu Wu, Hang Zhou, Mingsheng Long, and Jianmin Wang. Interpretable weather forecasting for worldwide stations with a unified deep model. *Nature Machine Intelligence*, 5(6):602–611, 2023b.

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.

Weijia Zhang, Le Zhang, Jindong Han, Hao Liu, Yanjie Fu, Jingbo Zhou, Yu Mei, and Hui Xiong. Irregular traffic time series forecasting based on asynchronous spatio-temporal graph convolutional networks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4302–4313, 2024a.

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

Zexi Zhang, Lifan Zhao, Yanyan Shen, and Bin Yao. Stockcl: Selective contrastive learning for stock trend forecasting via learnable concepts. In *International Conference on Database Systems for Advanced Applications*, pp. 275–284. Springer, 2024b.

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. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11106–11115, 2021.

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*, pp. 27268–27286, 2022.

A PROOFS

Throughout, $\mathbf{A}^{(l)}$ is evaluated without Gumbel noise, as at inference.

Preliminary. $\mathbf{H}^{(1)} = \mathbf{H}$ depends on $\mathbf{X}$ alone. If $\mathbf{H}^{(l)}$ depends on $\mathbf{X}$ alone and $l < \ell$, then $\mathbf{A}^{(l)}$, $\mathbf{K}^{(l)}$, $\mathbf{V}^{(l)}$ and the unmodulated $\mathbf{Q}^{(l)}$ are functions of $\mathbf{H}^{(l)}$, and so is $\mathbf{H}^{(l+1)}$. By induction, $\mathbf{H}^{(\ell)}$, $\mathbf{A}^{(\ell)}$, $\bar{\mathbf{K}}^{(\ell)} = \mathbf{A}^{(\ell)\top}\mathbf{K}^{(\ell)}$ and $\bar{\mathbf{V}}^{(\ell)} = \mathbf{A}^{(\ell)\top}\mathbf{V}^{(\ell)}$ depend on $\mathbf{X}$ alone. We drop the superscript (ℓ) below.

Proof of Proposition 1. If $\mathbf{Q} = \mathbf{H}\mathbf{W}_Q$, then $\mathbf{R} = \mathrm{softmax}(\mathbf{Q}\bar{\mathbf{K}}^\top/\sqrt{d})$ is a composition of functions of $\mathbf{X}$ and is therefore invariant to $\mathbf{C}$. For the modulated query, let two covariate tensors $\mathbf{C}$ and $\mathbf{C}'$ induce $(\boldsymbol{\gamma}_n, \boldsymbol{\beta}_n)$ and $(\boldsymbol{\gamma}'_n, \boldsymbol{\beta}'_n)$, and set $\boldsymbol{\delta}_n = (\boldsymbol{\gamma}_n - \boldsymbol{\gamma}'_n) \odot \mathbf{Q}_n + \boldsymbol{\beta}_n - \boldsymbol{\beta}'_n$. The logit of prototype j for variate n changes by $\boldsymbol{\delta}_n^\top \bar{\mathbf{K}}_j/\sqrt{d}$. Since the softmax is invariant only to shifts that are constant in j, row n of the retrieval matrix changes if and only if $\boldsymbol{\delta}_n^\top(\bar{\mathbf{K}}_j - \bar{\mathbf{K}}_{j'}) \neq 0$ for some j, j', that is, whenever the covariates change the relative logits over the prototypes. □

Proof of Proposition 2. Row n of $\mathbf{H}' = \mathbf{R}\bar{\mathbf{V}}$ is $\sum_{j=1}^{r} \mathbf{R}_{n,j}\bar{\mathbf{V}}_j$. The weight $\mathbf{R}_{n,j} = \mathrm{softmax}_j(\tilde{\mathbf{Q}}_n\bar{\mathbf{K}}^\top/\sqrt{d})$ depends on $\mathbf{C}$ through $\tilde{\mathbf{Q}}_n$ and on $\mathbf{X}$ through $\mathbf{Q}_n$ and $\bar{\mathbf{K}}$, while $\bar{\mathbf{V}}_j = \sum_{n'} \mathbf{A}_{n',j}\mathbf{V}_{n'}$ depends on $\mathbf{X}$ alone by the preliminary. □

B DATASETS

The four benchmarks span 29 datasets. Table 5 lists the 26 datasets of the short-term, long-term and high-dimensional benchmarks, and Table 6 the three covariate datasets, which are described separately because they are the only ones that carry known covariates.

Covariate channels. All covariate channels of a dataset are stacked into one tensor and processed identically by the model. M5-cov has $M{=}9$ channels: the sell price, a price-availability flag and the SNAP indicator of the store's state, together with sine and cosine encodings of day of week and of month and two event-type indicators. Favorita has $M{=}7$: the `onpromotion` flag, the same calendar encodings, a national-holiday flag and a payday flag. FreshRetailNet-Daily has $M{=}9$: the discount, an activity flag, precipitation, average temperature, the calendar encodings and a holiday flag. A cyclic calendar field of period P is encoded as $(\sin(2\pi v/P), \cos(2\pi v/P))$; the remaining fields are binary indicators or real values.

Splits. Training, validation and test segments are divided in chronological order for every dataset, so that no future observation is available while fitting. The short-term datasets use a $60/20/20$ split, the four ETT datasets follow the twelve, four and four month convention of the long-term literature, and every remaining dataset, including the three covariate datasets, uses $70/10/20$. The validation and test segments each begin one look-back window early, so that the first target of a segment has a complete history without drawing targets from the preceding one. FreshRetailNet-Daily is short ($T{=}97$), so its validation segment contains only a few windows; its results should be read with this in mind.

Normalization. Normalization acts at two levels, neither of which reads the forecast target. A dataset-level standardizer is fitted on the training segment alone, with one mean and standard deviation per variate on the short-term, long-term and high-dimensional benchmarks, and a single pair for the whole target matrix on the covariate benchmark, so the errors of Table 4 are measured in globally standardized units. Each look-back window is then standardized by its own mean and standard deviation and the forecast rescaled by the same pair, as in RevIN (Kim et al., 2021) without affine parameters. Continuous series-specific covariates are standardized with training statistics, while binary indicators and the sine and cosine calendar encodings are left unchanged. For the short-term datasets we forecast the traffic flow channel, so one variate is one sensor.

Data sources. The short-term files are the PEMS release distributed with SCINet (Liu et al., 2022). The ETT files come from the ETDataset repository and the remaining long-term files from the

Table 5: Datasets of the short-term, long-term and high-dimensional benchmarks. N is the number of variates the model is built with. The high-dimensional group is the Time-HD suite released with U-Cast (Ni et al., 2025), in which Solar denotes a $5{,}162$-variate collection of NREL plants, distinct from the 137-variate Solar of the long-term group.

Benchmark	Dataset	N	Domain
Short-term	PEMS03	358	California freeway traffic flow
	PEMS04	307	California freeway traffic flow
	PEMS07	883	California freeway traffic flow
	PEMS08	170	California freeway traffic flow
Long-term	ETTh1	7	Electricity transformer, hourly
	ETTh2	7	Electricity transformer, hourly
	ETTm1	7	Electricity transformer, 15-minute
	ETTm2	7	Electricity transformer, 15-minute
	Weather	21	Meteorological indicators
	Solar	137	Photovoltaic plant output
	Electricity	321	Household electricity consumption
High-dimensional	Air	1,105	Air quality monitoring, China
	Measles	1,161	Measles case counts, England
	SP500	1,475	S&P 500 constituent prices
	Atec	1,569	Transaction volume
	Neurolib	2,000	Simulated neural activity
	Traffic-GBA	2,229	Bay Area freeway traffic
	Meter	2,898	Smart meters, London
	SIRS	2,994	Simulated epidemic compartments
	M5	3,049	Retail unit sales
	Wind	3,696	Global wind speed
	Temp	3,774	Global temperature
	Solar	5,162	NREL solar power plants
	Mobility	5,826	Community mobility reports
	Traffic-CA	7,491	California freeway traffic, sensor level
	Wiki-20k	20,000	Wikipedia page view counts

Table 6: Covariate datasets. T is the length of the series, N the number of variates, and L and S the look-back and horizon used in Table 4. One variate is one store–SKU series. Covariates are defined on the same time index as the target, so a window $[t, t{+}L{+}S)$ exposes them over the forecast horizon as well as the look-back; future targets are never inputs. Series-specific covariates vary across variates, while calendar-type covariates are common to all variates and are copied along the variate axis.

Dataset	T	N	L	S	Series-specific covariates	Common covariates
M5-cov	1,941	30,490	96	28	sell price, price availability, SNAP	calendar, events
Favorita	729	22,604	96	16	on-promotion	calendar, holiday, payday
FreshRetailNet-Daily	97	50,000	28	7	discount, activity, precipitation, temperature	calendar, holiday

bundle distributed with Autoformer (Wu et al., 2021) and iTransformer (Liu et al., 2024). The high-dimensional files are the Time-HD release accompanying U-Cast (Ni et al., 2025), and the variate counts in Table 5 are those of that release.

C BASELINES

The short-term and long-term tables compare against DUET (Qiu et al., 2025), CycleNet (Lin et al., 2024), TimeXer (Wang et al., 2024), SOFTS (Han et al., 2024), iTransformer (Liu et al., 2024), PatchTST (Nie et al., 2023), Crossformer (Zhang & Yan, 2023), TimesNet (Wu et al., 2023a), DLinear (Zeng et al., 2023) and SCINet (Liu et al., 2022). The high-dimensional table compares against U-Cast (Ni et al., 2025), the strongest published method on this suite, together with DUET, PAttn (Tan et al., 2024), iTransformer, TSMixer (Chen et al., 2023), PatchTST, DLinear and TimesNet.

The covariate table compares against methods that consume known covariates by construction, TiDE (Das et al., 2023), TFT (Lim et al., 2021) and TSMixerX (Chen et al., 2023), and against three retrofits in which a covariate MLP (CovMLP) is added to a backbone that has no covariate path of its own. The three retrofits share one CovMLP, so that the comparison between them reflects the backbone rather than the fusion module. Every method in a given table receives the same covariates, the same splits and the same look-back and horizon, so the comparison isolates the architecture.

D METRICS

For a horizon of S steps over N variates, with $y_{s,n}$ the observation and $\hat{y}_{s,n}$ the point forecast,

$$\mathrm{MSE} = \frac{1}{NS}\sum_{s=1}^{S}\sum_{n=1}^{N}\left(y_{s,n}-\hat{y}_{s,n}\right)^2, \qquad \mathrm{MAE} = \frac{1}{NS}\sum_{s=1}^{S}\sum_{n=1}^{N}\left|y_{s,n}-\hat{y}_{s,n}\right|, \tag{12}$$

both computed in the standardized space in which the model is trained, and averaged over all test windows. The point forecast of SLIM is the $q{=}0.5$ output of the quantile head throughout.

E COMPLEXITY OF SLIM

We count multiply-accumulate operations for one layer and one sample, in the single-head form of Section 3, and drop constants and softmax nonlinearities. M denotes the number of covariate channels.

Cross-variate interaction. Table 7 follows the forward pass step by step. The three projections $\mathbf{Q},\mathbf{K},\mathbf{V} = \mathbf{H}\mathbf{W}_{\{Q,K,V\}}$ act on each variate independently and cost $O(Nd^2)$. They are common to any attention variant, including full cross-variate attention, so they do not enter the comparison between the two. Every remaining step is a product with either $\mathbf{A}\in\mathbb{R}^{N\times r}$ or a pooled matrix in $\mathbb{R}^{r\times d}$, and each contributes $O(Nrd)$:

$$\underbrace{O(Nrd)}_{\tilde{\mathbf{A}}=\mathrm{Linear}(\mathbf{H})} + \underbrace{O(Nrd)}_{\bar{\mathbf{K}}=\mathbf{A}^\top\mathbf{K}} + \underbrace{O(Nrd)}_{\bar{\mathbf{V}}=\mathbf{A}^\top\mathbf{V}} + \underbrace{O(Nrd)}_{\tilde{\mathbf{Q}}\bar{\mathbf{K}}^\top} + \underbrace{O(Nrd)}_{\mathrm{softmax}(\cdot)\,\bar{\mathbf{V}}} = O(Nrd). \tag{13}$$

Full cross-variate attention replaces the last two products by $\mathbf{Q}\mathbf{K}^\top$ and its application to $\mathbf{V}$, and forms no prototypes, so its interaction cost is $O(N^2d)$. The two differ by the factor r/N: for any fixed r, the prototype interaction is linear in N.

Covariate path. The covariate encoder compresses the time axis of each variate's covariates with a learned basis, projects the coefficients to width d, and emits a scale and a shift:

$$\underbrace{O\big(NMk(L+S)\big)}_{\mathbf{c}_n=\mathrm{Linear}_k(\mathbf{C}_{:,n,:})} + \underbrace{O(NMkd)}_{\mathbf{u}_n=\mathrm{Linear}(\mathbf{c}_n)} + \underbrace{O(Nd^2)}_{[\boldsymbol{\gamma}_n;\boldsymbol{\beta}_n]=\mathrm{Linear}(\mathbf{u}_n)} + \underbrace{O(Nd)}_{\tilde{\mathbf{Q}}_n=(1+\boldsymbol{\gamma}_n)\odot\mathbf{Q}_n+\boldsymbol{\beta}_n}. \tag{14}$$

Every term is linear in N, because the path contains only per-variate projections and no product between two variate-indexed matrices, and no term contains r. The prototype budget can therefore be changed without changing the cost of conditioning. Because the modulation is applied at the final layer only, this cost is incurred once per forward pass and does not grow with the depth ℓ.

Table 7: Cost of one SLIM layer for one sample. The last row is not part of SLIM; it is full cross-variate attention, given as the reference point for the comparison in the text. The projection row is shared by both and is excluded from the two interaction totals.

Step	Operation	Shape	Cost
Projections	$\mathbf{H}\mathbf{W}_{\{Q,K,V\}}$	$(N\times d)(d\times d)$	$O(Nd^2)$
Assignment	$\mathrm{Linear}(\mathbf{H})$	$(N\times d)(d\times r)$	$O(Nrd)$
Key pooling	$\mathbf{A}^\top\mathbf{K}$	$(r\times N)(N\times d)$	$O(Nrd)$
Value pooling	$\mathbf{A}^\top\mathbf{V}$	$(r\times N)(N\times d)$	$O(Nrd)$
Scores	$\tilde{\mathbf{Q}}\bar{\mathbf{K}}^\top$	$(N\times d)(d\times r)$	$O(Nrd)$
Retrieval	$\mathrm{softmax}(\cdot)\,\bar{\mathbf{V}}$	$(N\times r)(r\times d)$	$O(Nrd)$
Total interaction cost			
Full attention *(reference)*	$\mathbf{Q}\mathbf{K}^\top$ and its application to $\mathbf{V}$		$O(N^2d)$
SLIM	**sum of the five rows above**		$\boldsymbol{O(Nrd)}$

Memory. The interaction stores $\mathbf{A}$ and the retrieval weights, both $N \times r$, and the pooled $\bar{\mathbf{K}}, \bar{\mathbf{V}} \in \mathbb{R}^{r\times d}$, for $O(Nr + rd)$ activations. Full cross-variate attention stores an $N \times N$ score matrix, for $O(N^2)$. This is the asymptotic form of the gap measured in Section 4.5.

Scope. The counts assume a fixed prototype budget. If r grows with the variate count, the interaction cost $O(Nrd)$ grows with it and the bottleneck is no longer linear in N. Section 4.6 varies r on a fixed benchmark and reports the effect on accuracy and cost.

F Implementation Details

Protocol. Short-term results use L=96 and average horizons $S \in \{12, 24, 48, 96\}$; long-term results use L=96 and average $S \in \{96, 192, 336, 720\}$. The efficiency comparison uses L=96, S=7, and the SP500 prototype sweep uses L=336, S=7. Hyperparameters are selected on validation data only, and the selected configuration is evaluated once on the test split.

Model and optimization. Patches have length p=14 and stride s=7. The assignment temperature is parameterized as $\tau = \exp(\theta)$ with θ initialized at zero, so τ starts at one, is learned, and is clamped to $[0.1, 10]$. The Gumbel noise scale is fixed at α=0.15, and the noise is added during training only. Dropout is 0.1. Training uses Adam with a linear warmup followed by cosine decay, gradient-norm clipping at 1.0, at most 100 epochs, and early stopping with patience 15 on the validation split.

Quantile head. Every reported run predicts $|\mathcal{Q}|$=9 levels, $\mathcal{Q} = \{0.1, \ldots, 0.9\}$, and trains with the pinball loss. At inference the outputs are sorted along the quantile axis. Every MSE and MAE in this paper is computed from the q=0.5 output.

Hardware and reproducibility. All runs used NVIDIA RTX PRO 6000 Blackwell GPUs. Every configuration is fixed on validation data before test evaluation.

G SHOWCASE

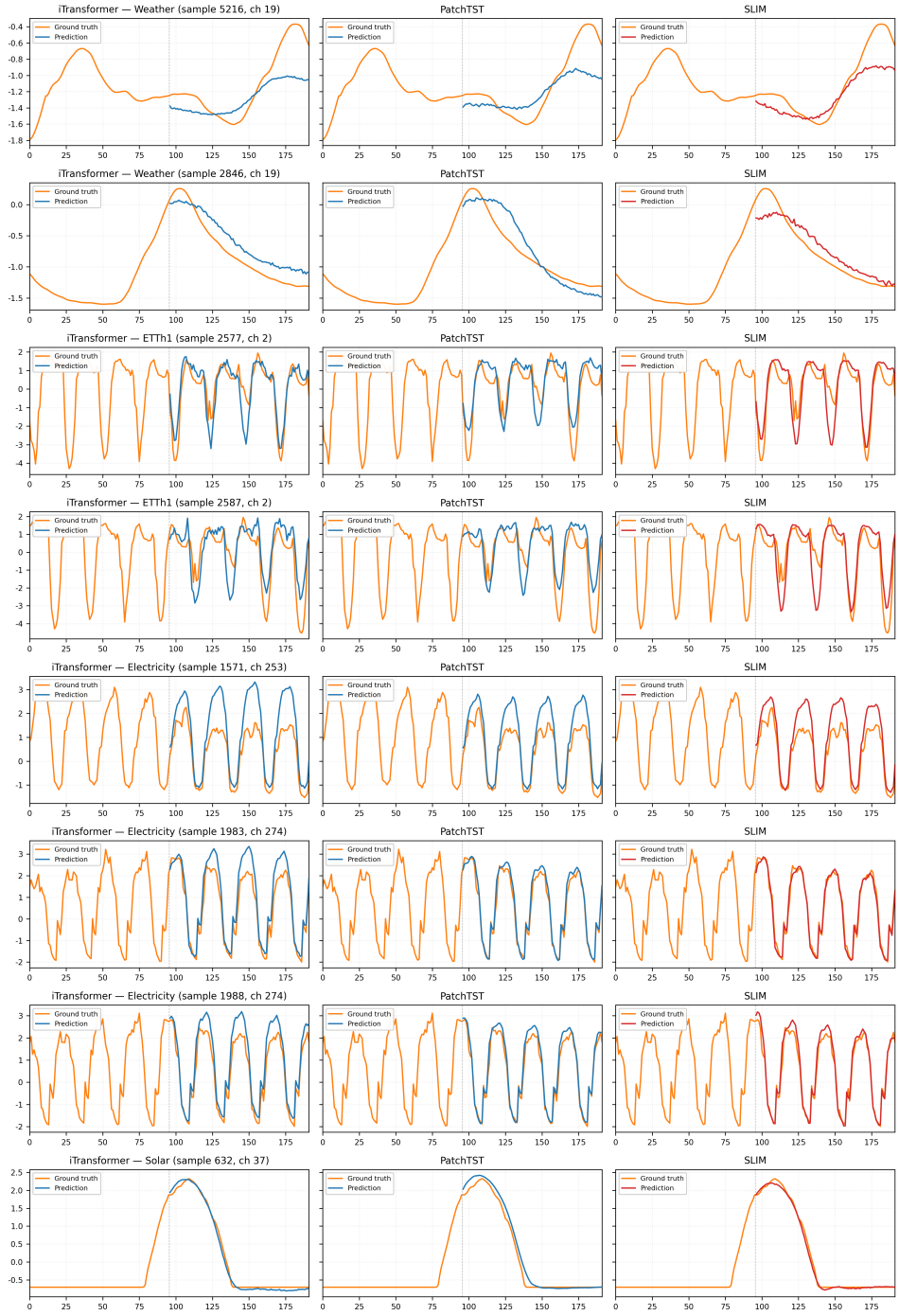

Figure 5: Forecast-blind validation examples from ETTm1 (N=7), Electricity (N=321) and PEMS07 (N=883). Gray denotes the history, black the ground truth, blue the median, and the band the central 80% interval. The displayed sample and variate maximize a data-only score combining target variance, target smoothness and a seasonal-repeat penalty; model forecasts are not used in selection. Panel MSE and 80% coverage are $0.330/86.9\%$, $0.00669/95.8\%$ and $0.00455/81.2\%$, respectively.

H FULL RESULTS

Tables 8 and 9 report the per-horizon results behind the averages in Tables 1 and 2. Both use a look-back of L=96.

Table 8: Per-horizon short-term multivariate time series forecasting results. Input length $L = 96$; results reported for $S \in \{12, 24, 48, 96\}$ on the four PEMS subsets and their average. Best and second-best highlighted in **bold** and underlined.

Models		SLIM Ours		DUET 2025		CycleNet 2024		TimeXer 2024		SOFTS 2024		iTransformer 2024		PatchTST 2023		Crossformer 2023		TimesNet 2023		DLinear 2023		SCINet 2022	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	12	**0.058**	**0.156**	0.064	0.166	0.070	0.173	0.066	0.172	0.064	0.165	0.071	0.174	0.099	0.216	0.090	0.203	0.085	0.192	0.122	0.243	0.066	0.172
	24	**0.072**	**0.173**	0.081	0.186	0.092	0.194	0.089	0.201	0.083	0.188	0.093	0.201	0.142	0.259	0.121	0.240	0.118	0.223	0.201	0.317	0.085	0.198
	48	**0.088**	**0.191**	0.114	0.222	0.129	0.229	0.136	0.247	0.114	0.223	0.125	0.236	0.211	0.319	0.202	0.317	0.155	0.260	0.333	0.425	0.127	0.238
	96	**0.108**	**0.210**	0.153	0.273	0.157	0.261	0.182	0.282	0.156	0.264	0.164	0.275	0.269	0.370	0.262	0.367	0.228	0.317	0.457	0.515	0.178	0.287
	Avg.	**0.082**	**0.182**	0.103	0.212	0.112	0.214	0.118	0.225	0.104	0.210	0.113	0.222	0.180	0.291	0.169	0.282	0.146	0.248	0.278	0.375	0.114	0.224
PEMS04	12	**0.069**	**0.167**	0.079	0.181	0.078	0.186	0.074	0.178	0.074	0.176	0.078	0.183	0.105	0.224	0.098	0.218	0.087	0.195	0.148	0.272	0.073	0.177
	24	**0.081**	**0.178**	0.096	0.203	0.099	0.212	0.087	0.195	0.088	0.194	0.095	0.205	0.153	0.275	0.131	0.256	0.103	0.215	0.224	0.340	0.084	0.193
	48	**0.097**	**0.193**	0.114	0.226	0.133	0.248	0.110	0.214	0.110	0.219	0.120	0.233	0.229	0.339	0.205	0.326	0.136	0.250	0.355	0.437	0.099	0.211
	96	**0.113**	**0.206**	0.136	0.257	0.167	0.281	0.148	0.251	0.135	0.244	0.150	0.262	0.291	0.389	0.402	0.457	0.190	0.303	0.452	0.504	0.114	0.227
	Avg.	**0.090**	**0.186**	0.106	0.217	0.119	0.232	0.105	0.209	0.102	0.208	0.111	0.221	0.195	0.307	0.209	0.314	0.129	0.241	0.295	0.388	0.092	0.202
PEMS07	12	**0.049**	**0.138**	0.060	0.156	0.062	0.162	0.057	0.152	0.057	0.152	0.067	0.165	0.095	0.207	0.094	0.200	0.082	0.181	0.115	0.242	0.068	0.171
	24	**0.057**	**0.149**	0.073	0.172	0.086	0.192	0.079	0.179	0.073	0.173	0.088	0.190	0.150	0.262	0.139	0.247	0.101	0.204	0.210	0.329	0.119	0.225
	48	**0.065**	**0.156**	0.096	0.201	0.128	0.234	0.099	0.191	0.096	0.195	0.110	0.215	0.253	0.340	0.311	0.369	0.134	0.238	0.398	0.458	0.149	0.237
	96	**0.073**	**0.167**	0.131	0.255	0.176	0.268	0.107	0.205	0.120	0.218	0.139	0.245	0.346	0.404	0.396	0.442	0.181	0.279	0.594	0.553	0.141	0.234
	Avg.	**0.061**	**0.153**	0.090	0.196	0.113	0.214	0.086	0.182	0.086	0.184	0.101	0.204	0.211	0.303	0.235	0.315	0.124	0.226	0.329	0.395	0.119	0.217
PEMS08	12	**0.063**	**0.158**	0.072	0.168	0.082	0.185	0.075	0.176	0.074	0.171	0.079	0.182	0.168	0.232	0.165	0.214	0.112	0.212	0.154	0.276	0.087	0.184
	24	**0.077**	**0.169**	0.093	0.191	0.117	0.226	0.102	0.201	0.104	0.201	0.115	0.219	0.224	0.281	0.215	0.260	0.141	0.238	0.248	0.353	0.122	0.221
	48	**0.101**	**0.181**	0.123	0.217	0.169	0.268	0.158	0.248	0.164	0.253	0.186	0.235	0.321	0.354	0.315	0.355	0.198	0.283	0.440	0.470	0.189	0.270
	96	**0.141**	**0.196**	0.166	0.291	0.233	0.306	0.366	0.377	0.211	0.253	0.221	0.267	0.408	0.417	0.377	0.397	0.320	0.351	0.674	0.565	0.236	0.300
	Avg.	**0.096**	**0.176**	0.113	0.217	0.150	0.246	0.175	0.251	0.138	0.220	0.150	0.226	0.280	0.321	0.268	0.306	0.193	0.271	0.379	0.416	0.159	0.244

Table 9: Per-horizon long-term multivariate time series forecasting results. Look-back length $L = 96$; horizons $S \in \{96, 192, 336, 720\}$. Best and second-best highlighted in **bold** and underlined.

Models		SLIM Ours		DUET 2025		CycleNet 2024		TimeXer 2024		SOFTS 2024		iTransformer 2024		PatchTST 2023		Crossformer 2023		TimesNet 2023		DLinear 2023		SCINet 2022	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.373**	**0.391**	0.377	0.393	0.375	0.395	0.382	0.403	0.381	0.399	0.386	0.405	0.414	0.419	0.423	0.448	0.384	0.402	0.386	0.400	0.654	0.599
	192	**0.426**	**0.421**	0.429	0.425	0.436	0.428	0.429	0.435	0.435	0.431	0.441	0.436	0.460	0.445	0.471	0.474	0.436	0.429	0.437	0.432	0.719	0.631
	336	**0.462**	**0.441**	0.471	0.446	0.496	0.455	0.468	0.448	0.480	0.452	0.487	0.458	0.501	0.466	0.570	0.546	0.491	0.469	0.481	0.459	0.778	0.659
	720	**0.458**	**0.459**	0.496	0.480	0.520	0.484	0.469	0.461	0.499	0.488	0.503	0.491	0.500	0.488	0.653	0.621	0.521	0.500	0.519	0.516	0.836	0.699
	Avg.	**0.430**	**0.428**	0.443	0.436	0.457	0.441	0.437	0.437	0.449	0.443	0.454	0.448	0.469	0.455	0.529	0.522	0.458	0.450	0.456	0.452	0.747	0.647
ETTh2	96	**0.286**	**0.334**	0.296	0.345	0.298	0.344	**0.286**	0.338	0.297	0.347	0.297	0.349	0.302	0.348	0.745	0.584	0.340	0.374	0.333	0.387	0.707	0.621
	192	**0.355**	**0.381**	0.368	0.389	0.372	0.396	0.363	0.389	0.373	0.394	0.380	0.400	0.388	0.400	0.877	0.656	0.402	0.414	0.477	0.476	0.860	0.689
	336	**0.386**	**0.406**	0.411	0.422	0.431	0.439	0.414	0.423	0.410	0.426	0.428	0.432	0.426	0.433	1.043	0.731	0.452	0.452	0.594	0.541	1.000	0.744
	720	**0.388**	**0.418**	0.412	0.424	0.450	0.458	0.408	0.432	0.411	0.433	0.427	0.445	0.431	0.446	1.104	0.763	0.462	0.468	0.831	0.657	1.249	0.838
	Avg.	**0.354**	**0.385**	0.372	0.395	0.388	0.409	0.368	0.396	0.373	0.400	0.383	0.407	0.387	0.407	0.942	0.683	0.414	0.427	0.559	0.515	0.954	0.723
ETTm1	96	**0.296**	**0.349**	0.324	0.354	0.325	0.363	0.318	0.356	0.325	0.361	0.334	0.368	0.329	0.367	0.404	0.426	0.338	0.375	0.345	0.372	0.418	0.438
	192	**0.341**	**0.378**	0.369	0.379	0.366	0.382	0.362	0.383	0.375	0.389	0.377	0.391	0.367	0.385	0.450	0.451	0.374	0.387	0.380	0.389	0.439	0.450
	336	**0.390**	**0.393**	0.404	0.402	0.396	0.401	0.395	0.407	0.405	0.412	0.426	0.420	0.399	0.410	0.532	0.515	0.410	0.411	0.413	0.413	0.490	0.485
	720	**0.436**	0.442	0.463	0.437	0.457	**0.433**	0.452	0.441	0.466	0.447	0.491	0.459	0.454	0.439	0.666	0.589	0.478	0.450	0.474	0.453	0.595	0.550
	Avg.	**0.366**	**0.391**	0.390	0.393	0.386	0.395	0.382	0.397	0.393	0.402	0.407	0.410	0.387	0.400	0.513	0.495	0.400	0.406	0.403	0.407	0.485	0.481
ETTm2	96	**0.164**	0.253	0.174	0.255	0.166	**0.248**	0.182	0.266	0.180	0.261	0.180	0.264	0.175	0.259	0.287	0.366	0.187	0.267	0.193	0.292	0.286	0.377
	192	**0.220**	0.293	0.243	0.302	0.233	**0.291**	0.248	0.306	0.246	0.306	0.250	0.309	0.241	0.302	0.414	0.492	0.249	0.309	0.284	0.362	0.399	0.445
	336	**0.272**	**0.328**	0.304	0.341	0.293	0.330	0.312	0.346	0.319	0.352	0.311	0.348	0.305	0.343	0.597	0.542	0.321	0.351	0.369	0.427	0.637	0.591
	720	**0.358**	**0.383**	0.399	0.397	0.395	0.389	0.414	0.404	0.405	0.401	0.412	0.407	0.402	0.400	1.730	1.042	0.408	0.403	0.554	0.522	0.960	0.735
	Avg.	**0.254**	**0.314**	0.280	0.324	0.272	0.315	0.289	0.331	0.287	0.330	0.288	0.332	0.281	0.326	0.757	0.611	0.291	0.333	0.350	0.401	0.571	0.537
Weather	96	**0.151**	**0.193**	0.163	0.202	0.158	0.203	0.157	0.205	0.166	0.208	0.174	0.214	0.177	0.210	0.158	0.230	0.172	0.220	0.196	0.255	0.221	0.306
	192	**0.202**	**0.244**	0.218	0.252	0.207	0.247	0.204	0.247	0.217	0.253	0.221	0.254	0.225	0.250	0.206	0.277	0.219	0.261	0.237	0.296	0.261	0.340
	336	**0.245**	**0.280**	0.274	0.294	0.262	0.289	0.261	0.290	0.282	0.300	0.278	0.296	0.278	0.290	0.272	0.335	0.280	0.306	0.283	0.335	0.309	0.378
	720	**0.318**	**0.332**	0.349	0.343	0.344	0.344	0.340	0.341	0.356	0.351	0.358	0.349	0.354	0.340	0.398	0.418	0.365	0.359	0.345	0.381	0.377	0.427
	Avg.	**0.229**	**0.262**	0.251	0.273	0.243	0.271	0.240	0.271	0.255	0.278	0.258	0.278	0.259	0.273	0.259	0.315	0.259	0.286	0.265	0.317	0.292	0.363
Solar	96	**0.182**	**0.203**	0.200	0.207	0.190	0.247	0.215	0.295	0.200	0.230	0.203	0.237	0.205	0.246	0.310	0.331	0.250	0.292	0.290	0.378	0.237	0.344
	192	**0.205**	**0.225**	0.228	0.233	0.210	0.266	0.236	0.301	0.229	0.253	0.233	0.261	0.237	0.267	0.734	0.725	0.296	0.318	0.320	0.398	0.280	0.380
	336	**0.210**	**0.231**	0.262	0.244	0.217	0.266	0.252	0.307	0.243	0.269	0.248	0.273	0.250	0.276	0.750	0.735	0.319	0.330	0.353	0.415	0.304	0.389
	720	**0.221**	**0.229**	0.258	0.249	0.223	0.266	0.244	0.305	0.245	0.272	0.249	0.275	0.252	0.275	0.769	0.765	0.338	0.337	0.356	0.413	0.308	0.388
	Avg.	**0.205**	**0.222**	0.237	0.233	0.210	0.261	0.237	0.302	0.229	0.256	0.233	0.262	0.236	0.266	0.641	0.639	0.301	0.319	0.330	0.401	0.282	0.375
Electricity	96	**0.130**	**0.226**	0.145	0.233	0.136	0.229	0.140	0.242	0.143	0.233	0.148	0.240	0.181	0.270	0.219	0.314	0.168	0.272	0.197	0.282	0.286	0.377
	192	**0.150**	0.249	0.163	0.248	0.152	**0.244**	0.157	0.256	0.158	0.248	0.162	0.253	0.188	0.274	0.231	0.322	0.184	0.289	0.196	0.285	0.399	0.445
	336	**0.163**	0.264	0.175	**0.262**	0.170	0.264	0.176	0.275	0.178	0.269	0.178	0.269	0.204	0.293	0.246	0.337	0.198	0.300	0.209	0.301	0.637	0.591
	720	0.209	0.313	**0.204**	**0.291**	0.212	0.299	0.211	0.306	0.218	0.305	0.225	0.317	0.246	0.324	0.280	0.363	0.220	0.320	0.245	0.333	0.960	0.735
	Avg.	**0.163**	0.263	0.172	**0.259**	0.168	**0.259**	0.171	0.270	0.174	0.264	0.178	0.270	0.205	0.290	0.244	0.334	0.193	0.295	0.212	0.300	0.571	0.537

I ADDITIONAL VISUALIZATIONS

Figure 6 visualizes the learned prototype assignment weights on PEMS04.

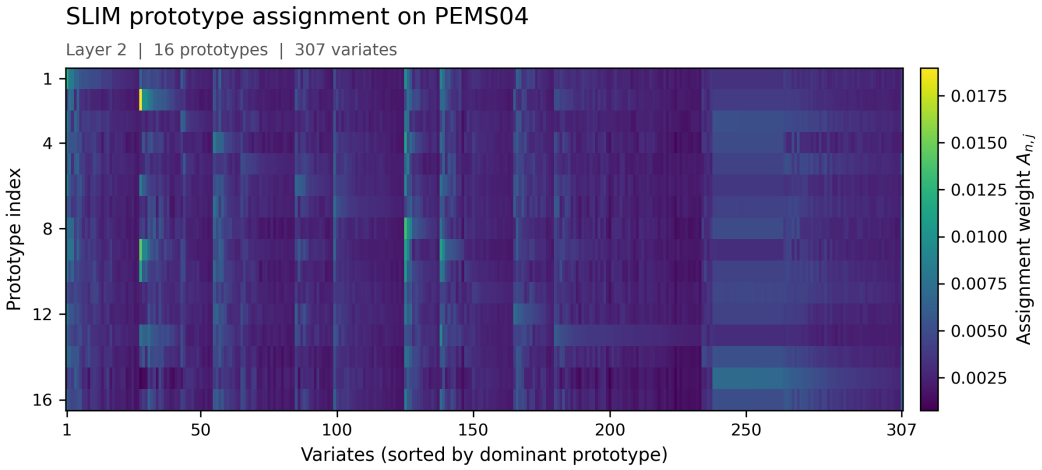

Figure 6: Prototype assignment weights on PEMS04 in the second layer of SLIM, with $r{=}16$ prototypes and $N{=}307$ variates. The heatmap displays $\mathbf{A}^\top$: rows correspond to prototypes, columns to variates, and color denotes $\mathbf{A}_{n,j}$. Columns are sorted by their dominant prototype. The weights of each prototype sum to one over the variates, as in Section 3.