SSVT: Industrial Multivariate Time Series Forecasting with Hybrid State Space and Variate-Token Attention

Anonymous authors
Paper under double-blind review

Abstract

As industrial systems operate at higher sampling rates and deploy increasingly dense sensor networks, multivariate time series forecasting in industrial settings faces two central challenges, namely slow cross-cycle drift that accumulates over long temporal spans and abrupt local fluctuations across variables. To address these challenges, we propose SSVT, a new paradigm for industrial multivariate time series forecasting that integrates hybrid state-space modeling with variate-token attention. Specifically, a temporal state-space encoder extracts cross-cycle historical dynamics for each variable and maps the resulting state representation into a corresponding variate token, allowing the tokens to retain long-term drift information along the temporal dimension. Variate-token attention then propagates information across variables and models local inter-variable dependencies, enabling SSVT to jointly capture long-term trends and short-term fluctuations. Building on this architecture, we further introduce physical constraints into the training process and use an LLM to assist in configuring the constraint weights. This design provides a more effective balance between data-driven learning and physical priors, improving forecasting stability and generalization in complex industrial environments. Experiments on six datasets show that the advantage of SSVT becomes increasingly pronounced as the forecasting horizon extends. In the 720-step forecasting task, SSVT achieves the best MSE and MAE across all six datasets. On the Boiler Equipment dataset, SSVT obtains an average MSE of 0.395, reducing the error by 6.95% compared with the strongest baseline. These results indicate that SSVT offers an effective paradigm for modeling cross-cycle slow drift and local variable dependencies in industrial time series. The code and experimental configurations will be made publicly available at https://github.com/CollaborativeResearch/SSVT.

1 Introduction

Industrial multivariate time series forecasting is a fundamental problem in operational monitoring and process control for complex equipment(Liu et al., 2025; Liu & Sun, 2025). Unlike general time series, industrial data are usually collected by multiple sensors during continuous operation. Their dynamics are affected by periodic operating conditions and control adjustments, and they also accumulate gradual changes as equipment ages(Qin et al., 2025). As a result, the main difficulty in industrial forecasting is not merely modeling long-range dependencies. The model must handle two types of non-stationary behavior that occur along different dimensions. The first is slow cross-cycle drift that gradually forms along the temporal dimension(Wei et al., 2024). The second is abrupt local fluctuations that propagate across variables(Liu et al., 2024a).

As shown in Figure 1, the same industrial variable may keep a similar local shape across multiple operating cycles, while its amplitude, baseline, or fluctuation range shifts during long-term operation. For example, equipment aging can make sensor sequences exhibit slow drift between adjacent cycles, and such drift is often difficult to explain from local patterns within a short window alone(Wang et al., 2024). Industrial variables are not independent.

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.

A change in operating condition may cause a local subset of variables to fluctuate synchronously or with short delays. Therefore, an effective forecasting model for industrial scenarios should provide two complementary capabilities. It should preserve cross-cycle historical states within each variable and model local inter-variable dependencies among variables.

Existing long-sequence forecasting methods address this problem from the perspectives of temporal modeling and variable interaction. Temporal patching, channel-independent modeling, and state-space models improve the efficiency of modeling long historical windows. In particular, selective state-space models represented by Mamba(Gu & Dao, 2024) encode long-sequence dynamics through recurrent states, avoiding the high computational cost of global self-attention along the temporal axis. However, these methods usually focus on the temporal evolution of each variable and may not fully capture local couplings among industrial sensors induced by process flows and control mechanisms(Yu et al., 2023; Sun et al., 2026). In contrast, variate-token methods such as iTransformer(Liu et al., 2024b) treat variables as attention units and explicitly enhance information propagation across variables. However, when each token is directly compressed from a raw historical window, temporal state extraction and variable relation modeling are mixed in the same representation(Xu et al., 2025; Wu et al., 2020). This coupling makes it difficult for the token representation to stably carry slow cross-cycle drift information. These observations suggest that industrial multivariate forecasting requires a modeling framework that separates temporal state encoding from variable-wise interaction, so that the model can handle both slow drift and local fluctuations.

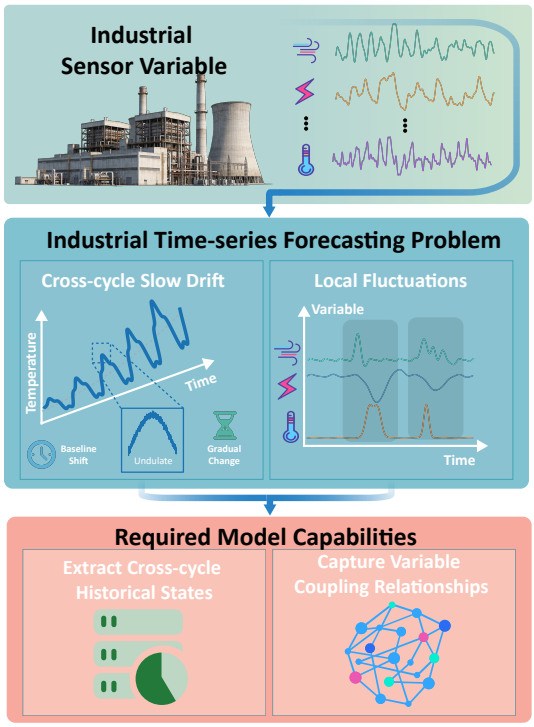


Figure 1: Cross-cycle slow drift and local fluctuations in an industrial variable.

Based on these observations, we propose SSVT, a forecasting framework for industrial multivariate time series that integrates hybrid state-space modeling with variate-token attention. SSVT models the two core types of industrial variation according to their dimensions. The Temporal State Space Encoder (TSSE) extracts cross-cycle historical states for each variable, capturing the slow drift that accumulates during long-term operation. The Variate-Token Attention (VTA) module performs information interaction along the variable dimension, capturing abrupt local couplings caused by disturbances such as operating-condition switches. In this way, SSVT decouples slow-drift modeling along the temporal dimension from abrupt-coupling modeling along the variable dimension. This design better matches the data generation process of industrial multi-sensor sequences. Furthermore, we incorporate industrial prior soft constraints into the training objective and use a large language model (LLM) to assist in configuring their weights. The LLM serves as an outer-loop configuration tool and does not participate in model prediction. The main contributions of this paper are summarized as follows:

- We propose SSVT, a state-encoded variable-token paradigm for industrial multivariate time-series forecasting. The proposed paradigm uses TSSE to extract cross-period historical dynamics for each variable and converts them into variable tokens, enabling cross-variable attention to operate on representations that contain long-term operating states.

- We introduce a dimension-specialized hybrid modeling architecture, where TSSE captures slow cross-period drift along the temporal dimension, while VTA models local and abrupt interactions along the variable dimension. This design decouples temporal state modeling from variable relation modeling, avoiding global self-attention over the full time axis while preserving local coupling information in industrial multi-sensor sequences.
- We incorporate soft industrial prior constraints into the training objective and design an LLM-assisted outer-loop mechanism for configuring constraint weights. The physical priors improve the consistency between predictions and operational laws, while the LLM is used to reduce manual trial and error in constraint-weight configuration and does not participate in the core forecasting process.
- We conduct experiments on the Boiler Equipment dataset and five public long-sequence forecasting benchmarks. The results show that SSVT achieves more stable advantages under long prediction horizons, especially in the 720-step forecasting setting. Ablation studies further verify the complementary roles of temporal state-space encoding and variable-token attention in handling cross-period slow drift and local variable interactions.

2 Related Work

2.1 Long-Sequence Dynamics and Slow Drift Modeling

Long-sequence forecasting methods mainly focus on capturing trends, periods, and long-range dependencies from historical windows(Woo et al., 2024). Early recurrent neural networks, convolutional neural networks, and their hybrid variants can describe local temporal evolution, but they often suffer from memory decay or limited receptive fields when the forecasting horizon becomes long(Ma et al., 2025; Wang et al., 2025). Methods based on the Transformer(Vaswani et al., 2017) architecture improve long-range dependency modeling through self-attention. Informer(Zhou et al., 2021), Autoformer(Wu et al., 2021), and FEDformer(Zhou et al., 2022) further introduce sparse attention, series decomposition, or frequency- domain modeling to reduce the computational cost of long-sequence forecasting. PatchTST(Nie et al., 2023) improves forecasting efficiency through temporal patching and channel-independent modeling, while linear models such as DLinear(Zeng et al., 2023) show that trend decomposition remains competitive on some tasks. However, slow drift in industrial scenarios is often caused by the long-term accumulation of equipment aging, operating-condition migration, and external disturbances(Liang et al., 2026; Shi et al., 2026). It does not always appear as regular periodicity or stationary trends. Normalization and de-stationarization methods can alleviate distribution shifts, but they do not explicitly preserve the historical state of each variable over long temporal scales. State-space models provide a feasible path for efficient long-history encoding by modeling sequence evolution through recurrent states. Nevertheless, if the encoding is performed independently for each variable, these models still have limited ability to use dependencies among variables.

2.2 Variable Interaction and Local Fluctuation Modeling

Local fluctuations in industrial multivariate time series usually occur within a limited set of related variables(Fan et al., 2024). For example, a load change may affect temperature, pressure, flow rate, and energy consumption variables that are connected by physical coupling or control mechanisms(Seo & Lim, 2025). To model such relations, LSTNet(Lai et al., 2018) uses convolutional structures to extract local variable patterns. Graph neural network(Feng et al., 2025) methods describe sensor dependencies through predefined or adaptive graph structures, and Crossformer(Zhang & Yan, 2023) explicitly models cross-dimensional relations. More recently, iTransformer(Liu et al., 2024b) represents the historical sequence of each variable as a variate token and performs attention along the variable dimension, providing a different token organization for multivariate forecasting. These methods show that variable interaction is important for multivariate forecasting, but they still face difficulty in balancing historical dynamic quality with local dependency modeling. Channel-independent

methods can ignore couplings among sensors, while global interaction may introduce redundant dependencies and additional computational cost. For variate-token attention, whether each token sufficiently contains historical dynamics directly affects the quality of cross-variable interaction. Based on this observation, this paper uses TSSE to extract historical states for each variable and then uses VTA to model local variable dependencies. Physical constraints and LLM-assisted weight configuration are further incorporated as training-stage mechanisms.

3 Method

3.1 Problem Formulation and Method Overview

We study industrial multivariate long-sequence forecasting. Given a historical window of length L, the model predicts the next H steps for C variables. Let $X \in \mathbb{R}^{B\times L\times C}$, $Y \in \mathbb{R}^{B\times H\times C}$, and $\widehat{Y} = f_\theta(X) \in \mathbb{R}^{B\times H\times C}$ denote the input, ground-truth future, and model prediction, respectively, where B is the batch size and f_θ is the forecasting model. Industrial sequences contain both slow cross-cycle drift and abrupt local fluctuations. Applying self-attention to the complete time-variable matrix increases the cost for long windows and mixes these two types of dependencies.

We therefore propose SSVT, which combines temporal state-space encoding with variate-token attention. After instance normalization, the model rearranges each variable history as a univariate sequence. TSSE extracts temporal states for each variable and converts the final hidden state into a state-encoded variate token. VTA then performs multi-head attention along the variate dimension before the prediction head produces the future sequence. As shown in Figure 2, SSVT separates temporal state encoding from variate interaction. TSSE captures slow drift along the temporal dimension, while VTA models local dependencies and abrupt co-fluctuations along the variate dimension.

3.2 TSSE

Industrial variables differ in units, mean levels, and fluctuation ranges. We therefore apply instance normalization independently to each sample and variable. Specifically,

$$\begin{aligned}
\mu_{b,c} &= \frac{1}{L}\sum_{t=1}^{L} X_{b,t,c}, \\
\sigma_{b,c} &= \sqrt{\frac{1}{L}\sum_{t=1}^{L}\left(X_{b,t,c}-\mu_{b,c}\right)^2+\varepsilon}, \\
\widetilde{X}_{b,t,c} &= \frac{X_{b,t,c}-\mu_{b,c}}{\sigma_{b,c}},
\end{aligned} \tag{1}$$

where b, c, and t index the sample, variable, and time step. The constant ε prevents division by zero. The saved μ and σ restore the prediction to the scale of the input window. When dataset-level standardization is also used, physical losses are computed after restoring predictions to the original engineering units.

The normalized input is permuted and reshaped into a collection of univariate sequences:

$$U = \mathrm{reshape}\Big(\mathrm{permute}(\widetilde{X})\Big) \in \mathbb{R}^{(B\cdot C)\times L\times 1}. \tag{2}$$

Variables do not exchange information before TSSE. A linear embedding and a learnable temporal positional representation then produce

$$H^0 = \mathrm{Emb}(U) + P_L, \qquad H^0 \in \mathbb{R}^{(B\cdot C)\times L\times D}. \tag{3}$$

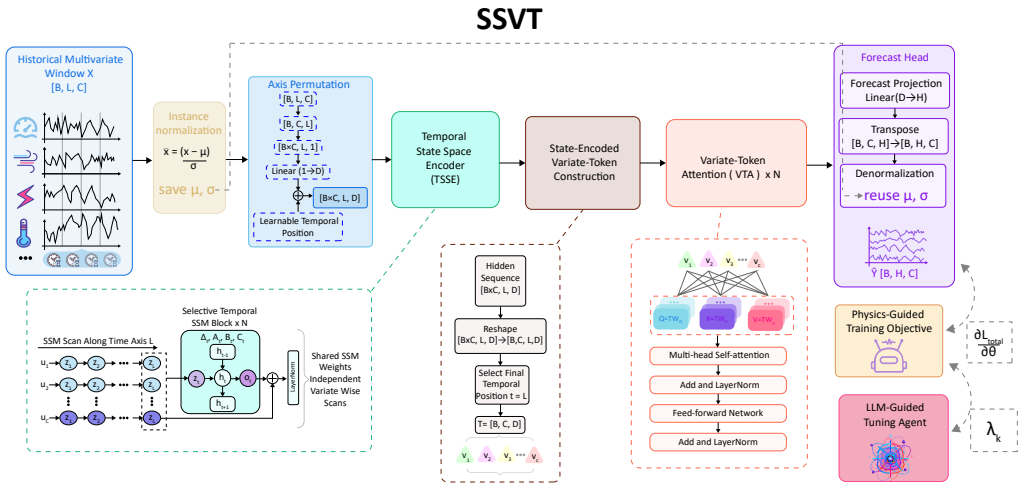


Figure 2: Overview of SSVT.

TSSE consists of N_t selective state-space blocks. For block l, the residual update is written as

$$\begin{aligned} \overline{H}^{\,l} &= \mathrm{SB}_l\left(H^{l-1}\right), \\ H^{l} &= \mathrm{LN}\Big(H^{l-1} + \mathrm{Dropout}\Big(\overline{H}^{\,l}\Big)\Big), \end{aligned} \tag{4}$$

where SB_l denotes the selective state-space block and LN denotes layer normalization. The state-space scan is performed inside SB_l, while the residual normalization improves training stability. We use the standard state update of the selected SSM and do not redesign its internal equations. All variables share TSSE parameters, keeping the temporal encoder size independent of the number of variables. TSSE encodes each variable without global self-attention over the temporal axis. Its output is used to represent the slow cross-cycle drift accumulated during long-term operation.

3.3 State-Encoded Variate-Token Construction

The TSSE output is reshaped as

$$\widetilde{H}^{N_t} = \mathrm{reshape}\big(H^{N_t}\big) \in \mathbb{R}^{B\times C\times L\times D}, \qquad H^{N_t} \in \mathbb{R}^{(B\cdot C)\times L\times D}. \tag{5}$$

Because the state-space encoder progressively aggregates information along the temporal axis, we use the final hidden state as the token for each variable:

$$T_{b,c,d} = \widetilde{H}^{N_t}_{b,c,L,d}, \qquad d = 1,\dots,D, \qquad T \in \mathbb{R}^{B\times C\times D}. \tag{6}$$

Thus, $T_{b,c}$ already contains the temporal state of variable c before variate interaction. SSVT therefore compresses the temporal dimension into tokens along the variate dimension, allowing the next attention layers to focus on cross-variate information exchange.

3.4 VTA

Let $T^{a-1} \in \mathbb{R}^{B\times C\times D}$ be the input to the a-th VTA layer. VTA treats variables as the attention sequence. The query, key, and value representations are computed as

$$\begin{aligned} Q^a &= T^{a-1}W_Q^a, \\ K^a &= T^{a-1}W_K^a, \\ V^a &= T^{a-1}W_V^a, \end{aligned} \tag{7}$$

where W_Q^a, W_K^a, and W_V^a are learnable projection matrices. The token representations are then updated by multi-head self-attention and a feed-forward network:

$$\begin{aligned} \widetilde{T}^a &= \mathrm{LN}\big(T^{a-1} + \mathrm{Dropout}(\mathrm{MHA}(Q^a, K^a, V^a))\big)\,, \\ T^a &= \mathrm{LN}\Big(\widetilde{T}^a + \mathrm{Dropout}\Big(\mathrm{FFN}(\widetilde{T}^a)\Big)\Big)\,. \end{aligned} \tag{8}$$

After N_a layers, the prediction head maps each variate token to an H-step trajectory:

$$\begin{aligned} \widehat{Y}_{\mathrm{BCH}} &= \mathrm{Linear}\big(T^{N_a}\big) \in \mathbb{R}^{B\times C\times H}, \\ \widehat{Y} &= \mathrm{Transpose}\Big(\widehat{Y}_{\mathrm{BCH}}\Big) \in \mathbb{R}^{B\times H\times C}. \end{aligned} \tag{9}$$

In the implementation, $N_a = \max(1, N_t)$. The two layer counts are kept distinct to emphasize their different temporal and variate-level roles. If instance normalization is enabled, the prediction is denormalized with the saved μ and σ. VTA models local variate dependencies and abrupt co-fluctuations without global temporal self-attention. Together, TSSE and VTA separate slow-drift modeling from local variate interaction.

3.5 Physics-Guided Training Objective

Forecasting accuracy alone may produce values outside normal operating ranges or with implausibly large changes. We therefore add soft industrial-prior constraints to the supervised objective. Let $g(\cdot)$ restore predictions to the original physical scale. $\mathcal{K}_d$ denotes the set of constraints enabled for dataset d. R_k represents the rule associated with constraint k. The total objective is

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{data}}\Big(\widehat{Y}, Y\Big) + \sum_{k\in\mathcal{K}_d} \lambda_k \mathcal{L}_{\mathrm{phys},k}\Big(g(\widehat{Y}); R_k\Big)\,. \tag{10}$$

The data loss is MSE by default, while λ_k controls the effect of each soft constraint. Physical losses are computed on $g(\widehat{Y})$ due to factors such as lag in response.

The constraints are auxiliary training losses and do not change the SSVT forward structure. For Boiler Equipment, we consider priors on the operating range. Smoothness is also encouraged. Distribution consistency is maintained, while lagged signs are respected. For public benchmarks with limited industrial semantics, we use a weaker smoothness prior when the variable meanings are interpretable. These are soft process regularizers rather than a strict PINN based on partial-differential-equation residuals.

3.6 LLM-Guided Outer-Loop Weight Tuning

The constraint weights determine the trade-off between forecasting error and physical consistency. To reduce manual trial and error, we use an LLM to adjust candidate weights across training rounds. After the r_{th} round, diagnostics $D^{(r)}$, the available rules R_{phys}, and the search bounds $\mathcal{B}_\lambda$ are provided as structured input. The proposed candidate is

$$\widetilde{\boldsymbol{\lambda}}^{(r+1)} = \mathrm{LLM}\Big(D^{(r)}, R_{\mathrm{phys}}, \mathcal{B}_\lambda\Big)\,. \tag{11}$$

Before training, the candidate is clipped and checked by deterministic numerical guardrails:

$$\boldsymbol{\lambda}^{(r+1)} = \mathrm{Guard}\Big(\mathrm{Clip}\Big(\widetilde{\boldsymbol{\lambda}}^{(r+1)}; \mathcal{B}_\lambda\Big)\Big)\,. \tag{12}$$

The guardrails ensure that fields are complete and configurations are unique. They further validate ranges and evaluate validation performance. Valid candidates enter the next training round. The search stops when the best physical configuration fails to improve over the no-constraint baseline or falls below a predefined improvement threshold. The final configuration is selected on the validation set, while the test set is used for final evaluation.

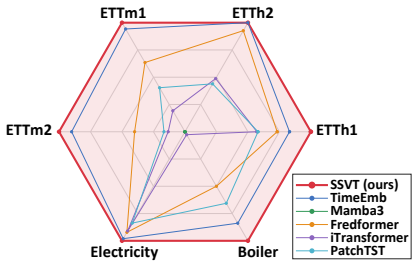
4 Experiments

4.1 Experimental Setup

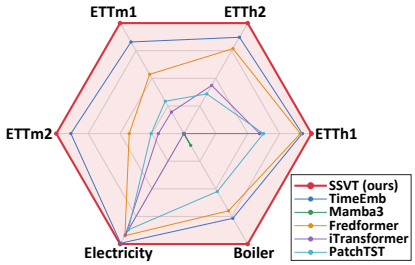
We evaluate SSVT on six multivariate forecasting datasets: the hourly ETTh1/ETTh2 and 15-min ETTm1/ETTm2 benchmarks (Zhou et al., 2021), Electricity (Trindade, 2015; Lai et al., 2018), and the industrial multi-sensor Boiler Equipment dataset (Hu et al., 2025). The latter assesses robustness to cross-cycle drift and local variate fluctuations, while the public benchmarks evaluate generalization. Forecasting horizons are set to 96, 192, 336, and 720 steps. We report mean squared error (MSE) and mean absolute error (MAE), with lower values indicating better accuracy. For Boiler Equipment, we additionally measure range, variation-magnitude, distribution, and lag-relation violations. SSVT is compared with TimeEmb(Xia et al., 2025), Mamba3(Lahoti et al., 2026), Fredformer(Piao et al., 2024), iTransformer(Liu et al., 2024b), and PatchTST(Nie et al., 2023). All methods use identical data splits, with hyperparameters selected on the validation set and final results reported on the test set. Constraint weights are selected using only the training and validation sets.

4.2 Main Forecasting Results

Figures 3(a) and 3(b) visualize the MSE and MAE trends across prediction horizons, respectively. Both metrics show how the relative advantage of SSVT changes as the forecasting horizon increases. Table 1 compares SSVT with five baselines on the six datasets. SSVT achieves the best MSE and MAE at the 720-step horizon on all datasets, suggesting that its advantage becomes clearer when the forecast must preserve long-term temporal states and cross-variate interactions. On Boiler Equipment, SSVT obtains an average MSE of 0.395, compared with 0.425 for TimeEmb, corresponding to a 6.95% reduction. Its average MAE is 0.428, which is 5.33% lower than the TimeEmb result of 0.452. SSVT also remains competitive on the public benchmarks. It achieves the lowest average MSE on ETTh1, ETTm1, ETTm2, and Electricity. On ETTh2, its average MSE is close to TimeEmb, while its average MAE is lower. The advantage is not uniform at short horizons, Fredformer performs better than SSVT on the 96-step Boiler Equipment task. This pattern indicates that SSVT is mainly beneficial for long-horizon forecasting rather than short-term local fitting.



(a) MSE



(b) MAE

Figure 3: Average forecasting errors across prediction horizons.

4.3 Ablation Study of the Hybrid Backbone

We study the roles of TSSE and VTA on ETTh1 and Boiler Equipment with the same data split, input length, forecasting horizon, and optimization configuration. All ablation experiments use $H = 720$. As shown in Table 2, "w/o VTA" keeps TSSE and the prediction head, "w/ Temporal Attn." replaces variate interaction with temporal attention, "w/o TSSE" constructs variate tokens without temporal state encoding, and "Full SSVT" contains both modules.

Full SSVT achieves the best results on both datasets. On ETTh1, it reduces MSE and MAE by approximately 16% and 12% relative to "w/o VTA", and by 7% and 6% relative to "w/o TSSE". On Boiler Equipment, Full SSVT obtains MSE 0.648 and MAE 0.608, clearly outperforming all variants. Adding temporal attention after TSSE does not improve

Table 1: Multivariate forecasting results. Each cell reports MSE / MAE. For each dataset and prediction horizon H, the best result for each metric is in red, and the second-best is in blue. MSE and MAE are ranked independently; lower is better. The Avg. row is also compared across models.

Dataset	Pred. Length	SSVT (ours)		TimeEmb		Mamba3		Fredformer		iTransformer		PatchTST	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.306	0.382	0.366	0.387	0.460	0.455	0.373	0.392	0.386	0.405	0.394	0.406
	192	0.413	0.417	0.417	0.416	0.518	0.484	0.433	0.420	0.441	0.436	0.440	0.435
	336	0.449	0.428	0.457	0.436	0.548	0.504	0.470	0.437	0.487	0.458	0.491	0.462
	720	0.457	0.453	0.459	0.460	0.545	0.512	0.467	0.456	0.503	0.491	0.487	0.479
	Avg.	0.406	0.420	0.425	0.425	0.518	0.489	0.436	0.426	0.454	0.448	0.453	0.446
ETTh2	96	0.279	0.330	0.277	0.328	0.313	0.363	0.293	0.342	0.297	0.349	0.288	0.340
	192	0.351	0.383	0.356	0.379	0.397	0.412	0.371	0.389	0.380	0.400	0.376	0.395
	336	0.401	0.402	0.400	0.417	0.449	0.457	0.382	0.409	0.428	0.432	0.440	0.451
	720	0.410	0.423	0.416	0.437	0.452	0.465	0.415	0.434	0.427	0.445	0.436	0.453
	Avg.	0.362	0.385	0.362	0.390	0.403	0.424	0.365	0.394	0.383	0.407	0.385	0.410
ETTm1	96	0.301	0.341	0.304	0.343	0.359	0.387	0.326	0.361	0.334	0.368	0.329	0.365
	192	0.352	0.370	0.354	0.373	0.401	0.407	0.363	0.380	0.377	0.391	0.380	0.394
	336	0.372	0.381	0.379	0.393	0.425	0.423	0.395	0.403	0.426	0.420	0.400	0.410
	720	0.434	0.418	0.435	0.428	0.483	0.453	0.453	0.438	0.491	0.459	0.475	0.453
	Avg.	0.365	0.377	0.368	0.384	0.417	0.418	0.384	0.396	0.407	0.410	0.396	0.406
ETTm2	96	0.167	0.244	0.163	0.242	0.184	0.271	0.177	0.259	0.180	0.264	0.184	0.264
	192	0.224	0.284	0.226	0.285	0.255	0.317	0.243	0.301	0.250	0.309	0.246	0.306
	336	0.279	0.316	0.286	0.324	0.317	0.355	0.302	0.340	0.311	0.348	0.308	0.346
	720	0.377	0.373	0.383	0.381	0.412	0.412	0.397	0.396	0.412	0.407	0.409	0.402
	Avg.	0.262	0.304	0.265	0.308	0.292	0.339	0.280	0.324	0.288	0.332	0.287	0.330
Electricity	96	0.136	0.238	0.136	0.231	0.297	0.374	0.147	0.241	0.148	0.240	0.164	0.251
	192	0.152	0.242	0.153	0.246	0.303	0.381	0.165	0.258	0.162	0.253	0.173	0.262
	336	0.167	0.266	0.170	0.264	0.317	0.391	0.177	0.273	0.178	0.269	0.190	0.279
	720	0.200	0.291	0.208	0.297	0.354	0.416	0.213	0.304	0.225	0.317	0.230	0.313
	Avg.	0.164	0.259	0.167	0.260	0.318	0.391	0.176	0.269	0.178	0.270	0.189	0.276
Boiler Equipment	96	0.102	0.183	0.140	0.220	0.405	0.406	0.081	0.175	0.175	0.268	0.177	0.259
	192	0.341	0.404	0.412	0.439	0.523	0.487	0.342	0.413	0.468	0.481	0.410	0.454
	336	0.489	0.516	0.495	0.527	0.617	0.547	0.694	0.541	0.707	0.620	0.548	0.555
	720	0.648	0.608	0.652	0.621	0.780	0.639	0.833	0.705	0.953	0.754	0.702	0.640
	Avg.	0.395	0.428	0.425	0.452	0.581	0.520	0.488	0.459	0.576	0.531	0.459	0.477

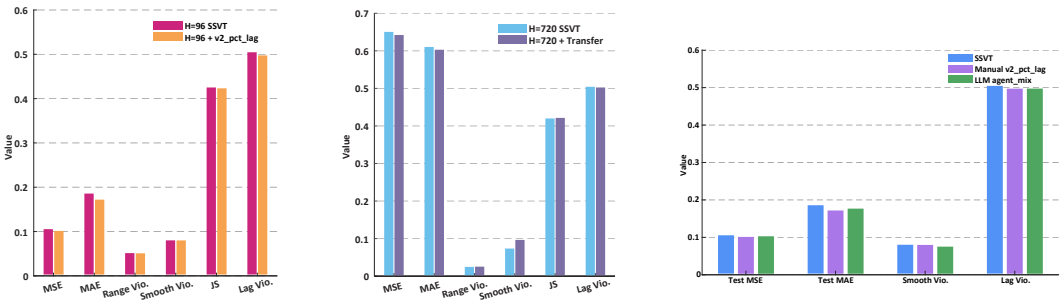
Table 2: Ablation study of the hybrid backbone at $H = 720$. For each metric, the best result is in red and the second-best is in blue; lower is better.

Variant	Setting	ETTh1 MSE	ETTh1 MAE	Boiler MSE	Boiler MAE
w/o VTA	TSSE-only	0.545	0.512	0.780	0.639
w/ Temporal Attn.	TSSE + temporal attention	0.566	0.528	0.779	0.638
w/o TSSE	VTA-only	0.492	0.484	0.779	0.639
Full SSVT	TSSE + VTA	0.457	0.453	0.648	0.608

the results, indicating that temporal interaction alone cannot replace explicit information exchange along the variate dimension.

4.4 Physics-Guided Constraints and Weight Configuration

We study the industrial-prior soft constraints on the Boiler Equipment dataset and then test whether an LLM can reduce the effort required to configure their weights. These priors are adjustable. They do not form a strict PINN. The range prior limits outlet-temperature deviations. The smoothness prior penalizes changes above the 95th percentile of training variation. The distribution prior uses Jensen–Shannon (JS) divergence. The lag prior encodes the weak negative relation between spray-water flow changes and outlet-temperature changes about 60 steps later. The broader study considers horizons of 96, 192, 336, and 720 steps. Figure 4 summarizes the two horizon settings and the weight-configuration search.



(a) $H = 96$ (b) $H = 720$ (c) Weight search

Figure 4: Effects of soft industrial-prior constraints and weight configuration on Boiler Equipment. Lower values indicate smaller forecasting errors or fewer constraint violations.

At $H = 96$, the manually selected v2_pct_lag reduces MSE from 0.1024 to 0.0984 and MAE from 0.1827 to 0.1690, corresponding to decreases of 3.98% and 7.48%, while also reducing range violation from 0.0483 to 0.0479, smoothness violation from 0.0771 to 0.0770, JS divergence from 0.4221 to 0.4202, and lag violation from 0.5016 to 0.4945 (Figure 4(a)). At $H = 720$, the transferred configuration lowers MSE from 0.6484 to 0.6401 and MAE from 0.6083 to 0.6009, corresponding to reductions of 1.28% and 1.23%, respectively, but increases range violation from 0.0227 to 0.0233, smoothness violation from 0.0715 to 0.0945, and JS divergence from 0.4179 to 0.4194, while slightly reducing lag violation from 0.5022 to 0.5004 (Figure 4(b)). For the $H = 96$ weight search, manual selection evaluates five candidates and obtains MSE 0.0984 and MAE 0.1690, where as the LLM proposes agent_mix after one trial and improves over unconstrained SSVT, reaching MSE 0.1001 and MAE 0.1740 without matching the manual configuration. The LLM achieves the lowest smoothness violation (0.0724), while its lag violation (0.4947) remains close to the manual result (0.4945) and below the unconstrained value (0.5016) (Figure 4(c)). These results indicate that soft constraints provide horizon-dependent accuracy–consistency trade-offs, while LLM assistance primarily reduces configuration effort rather than directly improving forecasting.

5 Conclusion

This paper focuses on industrial multivariate time series forecasting. It tackles slow cross-cycle drift. Local variate fluctuations are handled separately. To address there challages, we propose SSVT, which combines temporal state-space encoding with variate-token attention. TSSE extracts the cross-cycle historical state of each variable, while VTA models local dependencies and co-fluctuations along the variate dimension. This factorization separates temporal state encoding from variate interaction and matches the two dominant types of variation in industrial multi-sensor sequences. Experiments on six datasets and four forecasting horizons show that SSVT provides consistent performance improvements, especially at long horizons. At $H = 720$, SSVT achieves the best MSE and MAE on all six datasets. On Boiler Equipment, it obtains average MSE and MAE values of 0.395 and 0.428, respectively, reducing the errors over the best baseline by 6.95% and 5.33%. Ablation studies on ETTh1 and Boiler Equipment further show that the full SSVT backbone outperforms its TSSE-only, VTA-only, and temporal-attention variants. The industrial-prior soft constraints reduce MSE and MAE by 3.98% and 7.48% at $H = 96$, while the LLM-assisted configuration improves over the unconstrained baseline with fewer candidate trials. Both components are auxiliary training mechanisms rather than parts of the core predictor. Future work will study more stable industrial-prior modeling and adaptive forecasting under changing operating conditions.

LLM Usage Statement

Large language models were used solely to polish the writing and to assist with debugging code. They did not contribute to the research ideas, experimental design, or analysis. All results were verified by the authors, who take full responsibility for the content of this paper.

References

Xinyao Fan, Yueying Wu, Chang Xu, Yu-Hao Huang, Weiqing Liu, and Jiang Bian. MG-TSD: Multi-granularity time series diffusion models with guided learning process. In International Conference on Learning Representations, pp. 12916–12934, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/3792ac815260f4ba55ed71c7906cce36-Paper-Conference.pdf.

X. Feng, H. Li, and Y. Yang. Time-lagged relation graph neural network for multivariate time series forecasting. Engineering Applications of Artificial Intelligence, 139, 2025.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In First Conference on Language Modeling (COLM), 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Weijun Hu, Anze Jiang, Kaihong Chen, Jie Zheng, Wenli Shang, and Zhong Cao. A long-tailed distribution time-series dataset in boiler equipment. Scientific Data, 12(1):742, 2025. doi: 10.1038/s41597-025-05096-4.

Aakash Lahoti, Kevin Y. Li, Berlin Chen, Caitlin Wang, Aviv Bick, J. Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles. In International Conference on Learning Representations, 2026. URL https://iclr.cc/virtual/2026/poster/10010352.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 95–104, 2018. doi: 10.1145/3209978.3210006.

X. Liang, K. Hao, L. Chen, et al. Causal structure-based forecasting for multivariate industrial time series under covariate drift. Knowledge-Based Systems, 347, 2026.

Peiyuan Liu, Beiliang Wu, Yifan Hu, Naiqi Li, Tao Dai, Jigang Bao, and Shu-Tao Xia. TimeBridge: Non-stationarity matters for long-term time series forecasting. In Proceedings of the 42nd International Conference on Machine Learning, volume 267 of Proceedings of Machine Learning Research, pp. 37761–37791. PMLR, 2025. URL https://proceedings.mlr.press/v267/liu25ai.html.

Shuhan Liu and Wenqiang Sun. Knowledge- and data-driven prediction of blast furnace gas generation and consumption in iron and steel sites. Applied Energy, 390, 2025. doi: 10.1016/j.apenergy.2025.125819.

Xingheng Liu, Erling Lunde, Fabio Diehl, Aibo Zhang, Jørn Vatn, and Shen Yin. Degradation modeling under time-varying operating conditions: Inference and prognosis with particle filter. Reliability Engineering & System Safety, 245, 2024a. doi: 10.1016/j.ress.2024.109965.

Yong Liu, Teng Hu, Hua 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, 2024b. URL https://openreview.net/forum?id=JePfAI8fah.

Jiaming Ma, Binwu Wang, Qihe Huang, Guanjun Wang, Pengkun Wang, Zhengyang Zhou, and Yang Wang. MoFo: Empowering long-term time series forecasting with periodic pattern modeling. In Advances in Neural Information Processing Systems 38, pp. 94558–94589, 2025. doi: 10.52202/085713-2844.

Yu 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. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Xihao Piao, Zheng Chen, Taichi Murayama, Yasuko Matsubara, and Yasushi Sakurai. Fredformer: Frequency debiased transformer for time series forecasting. In Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 2400–2410, 2024. doi: 10.1145/3637528.3671928.

Yi Qin, Yihang Zhao, Junyu Qi, and Yongfang Mao. Spatial-temporal multi-sensor information fusion network with prior knowledge embedding for equipment remaining useful life prediction. Reliability Engineering & System Safety, 264, 2025. doi: 10.1016/j.ress.2025.111420.

Hyunwoo Seo and Chiehyeon Lim. ST-MTM: Masked time series modeling with seasonal-trend decomposition for time series forecasting. In Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1, pp. 1209–1220, 2025. doi: 10.1145/3690624.3709254.

X. Shi, K. Hao, X. Zeng, L. Chen, and H. Li. Spatial information bottleneck graph structure learning based multivariate time series prediction for industrial processes. Journal of Manufacturing Systems, 85:441–454, 2026. doi: 10.1016/j.jmsy.2026.01.017.

Ziyue Sun, Hu Xu, Yinlong Li, Wenhai Wang, and Xinggao Liu. A semantically guided multimodal graph neural network for process factor forecasting of industrial IoT systems. Expert Systems with Applications, 303, 2026. doi: 10.1016/j.eswa.2025.129032.

Artur Trindade. ElectricityLoadDiagrams20112014. UCI Machine Learning Repository, 2015. URL https://doi.org/10.24432/C58C86. Dataset. DOI: 10.24432/C58C86.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems 30, pp. 5998–6008, 2017. URL https://papers.nips.cc/paper/7181-attention-is-all-you-need.

Yihang Wang, Yuying Qiu, Peng Chen, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. Towards a general time series forecasting model with unified representation and adaptive transfer. In Proceedings of the 42nd International Conference on Machine Learning, volume 267 of Proceedings of Machine Learning Research, pp. 64127–64151. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25ci.html.

Zezhou Wang, Jian Hou, Jiantai Zhu, Liyuan Wang, and Zhongyi Cai. Stochastic degradation modeling and remaining useful lifetime prediction based on long short-term memory network. Measurement, 234, 2024. doi: 10.1016/j.measurement.2024.114803.

Yujie Wei, Ershun Pan, and Zhi-Sheng Ye. Condition monitoring based on corrupted multiple time series with common trends. Reliability Engineering & System Safety, 251, 2024. doi: 10.1016/j.ress.2024.110324.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Moirai: Unified training of universal time series forecasting transformers. In Proceedings of the 41st International Conference on Machine Learning, volume 235 of Proceedings of Machine Learning Research. PMLR, 2024. URL https://proceedings.mlr.press/v235/woo24m.html.

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 34, pp. 22419–22430, 2021. URL https://papers.nips.cc/paper/2021/hash/bcc0d400288793e8fe763e63ae837ac0-Abstract.html.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 753–763, 2020. doi: 10.1145/3394486.3403118.

Mingyuan Xia, Chunxu Zhang, Zijian Zhang, Hao Miao, Qidong Liu, Yuanshao Zhu, and Bo Yang. TimeEmb: A lightweight static-dynamic disentanglement framework for time series forecasting. In Advances in Neural Information Processing Systems 38, pp. 84868–84899, 2025. doi: 10.52202/085713-2560.

Zhi Xu, Qiang Wang, Brian Y. Lim, Shiliang Sun, and Yuhong Xiao. SDTNet: A scalable dynamic topology network for multivariate time series forecasting. IEEE Transactions on Neural Networks and Learning Systems, 36:11903–11918, 2025. doi: 10.1109/TNNLS.2024.3424811.

Chengqing Yu, Fei Wang, Zezhi Shao, Tao Sun, Lin Wu, and Yongjun Xu. DSformer: A double sampling transformer for multivariate time series long-term prediction. In Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, pp. 3062–3072, 2023. doi: 10.1145/3583780.3614851.

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.

Yunjun Zhang and Juncheng Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In International Conference on Learning Representations, 2023. URL https://openreview.net/forum?id=vSVLM2j9eie.

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. doi: 10.1609/aaai.v35i12.17325.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In Proceedings of the 39th International Conference on Machine Learning, volume 162 of Proceedings of Machine Learning Research, pp. 27268–27286. PMLR, 2022. URL https://proceedings.mlr.press/v162/zhou22g.html.

A Dataset and Preprocessing Details

A.1 Datasets

We evaluate SSVT on six multivariate time series datasets: ETTh1, ETTh2, ETTm1, ETTm2, Electricity, and Boiler Equipment. As shown in Table 3, the first five datasets are public benchmarks, while Boiler Equipment represents an industrial multi-sensor forecasting scenario.

Table 3: Summary of the datasets used in this study.

Dataset	Domain	Sampling	Variables	Reference
ETTh1	Electricity transformer	1 hour	7	(Zhou et al., 2021)
ETTh2	Electricity transformer	1 hour	7	(Zhou et al., 2021)
ETTm1	Electricity transformer	15 minutes	7	(Zhou et al., 2021)
ETTm2	Electricity transformer	15 minutes	7	(Zhou et al., 2021)
Electricity	Electricity consumption	1 hour	321	(Trindade, 2015; Lai et al., 2018)
Boiler Equipment	Industrial boiler	5 seconds	30	(Hu et al., 2025)

The ETT datasets contain transformer-related measurements sampled at hourly or 15-minute intervals. ETTh1 and ETTh2 use hourly observations, whereas ETTm1 and ETTm2 use 15-minute observations. Electricity contains the electricity consumption records of 321 clients. Boiler Equipment contains industrial boiler measurements, including temperature, pressure, flow, oxygen, vibration, and differential-pressure variables.

The Boiler Equipment data were collected from March 27, 2022 to April 1, 2022 at a five-second sampling interval. The dataset contains 86,400 observations and 30 variables. Approximately 8.6% of the observations correspond to abnormal operating states. During normal operation, the boiler outlet steam temperature is approximately within 530–545°C.

The Boiler Equipment data are described in the Scientific Data article "A long-tailed distribution time-series dataset in boiler equipment" and are available through Figshare (https://doi.org/10.6084/m9.figshare.28868849).

A.2 Forecasting Protocol

For each dataset, a historical window of length $L = 96$ is used to predict the following H observations. The forecasting horizons are $H \in \{96, 192, 336, 720\}$. Before preprocessing, the input and target windows in the original physical scale are denoted by $X^{\text{raw}} \in \mathbb{R}^{B\times L\times C}$ and $Y^{\text{raw}} \in \mathbb{R}^{B\times H\times C}$, respectively, where B is the batch size and C is the number of variables.

The four ETT datasets use the chronological benchmark partition implemented by the data loader. The first 12 months are used for training, the next 4 months for validation, and the final 8 months for testing. For ETTh1 and ETTh2, the corresponding timestamp intervals and numbers of observations are:

- training: 2016-07-01 00:00–2017-06-25 23:00, 8,640 observations;
- validation: 2017-06-26 00:00–2017-10-23 23:00, 2,880 observations;
- test: 2017-10-24 00:00–2018-06-20 23:00, 5,760 observations.

For ETTm1 and ETTm2, which contain four observations per hour, the corresponding splits contain 34,560, 11,520, and 23,040 observations, respectively. Their timestamp intervals are 2016-07-01 00:00–2017-06-25 23:45 for training, 2017-06-26 00:00–2017-10-23 23:45 for validation, and 2017-10-24 00:00–2018-06-20 23:45 for testing.

Boiler Equipment and Electricity use chronological custom partitions because they do not follow the ETT benchmark calendar. The cleaned Boiler Equipment file contains 86,400 records sampled every 5 seconds and is partitioned into 60,480, 8,640, and 17,280 records for training, validation, and testing, respectively. The corresponding timestamp intervals

are 2022-03-27 14:28:54–2022-03-31 02:28:49, 2022-03-31 02:28:54–2022-03-31 14:28:49, and 2022-03-31 14:28:54–2022-04-01 14:28:49.

Electricity contains 26,304 hourly records and is partitioned into 18,412, 2,632, and 5,260 records for training, validation, and testing. The corresponding timestamp intervals are 2016-07-01 02:00–2018-08-07 05:00, 2018-08-07 06:00–2018-11-24 21:00, and 2018-11-24 22:00–2019-07-02 01:00. Neither custom partition uses random shuffling. All windows are generated in chronological order. The target ranges of the three splits are disjoint. To construct the first validation and test windows, the data loader retains the preceding $L = 96$ observations as historical context. This boundary context is used only as input and is not included in the future target, so it does not introduce target leakage. The data loader uses a label length of 48 when constructing the target window. However, the proposed forecasting backbone directly maps the historical input window to the future horizon and does not use decoder inputs during forecasting. The validation set is used for early stopping, model selection, and selection of physics-constraint weights, including the candidate selection in the LLM-assisted outer-loop procedure. The test set is accessed only after all model and weight-selection decisions have been fixed and is used for final reporting.

A.3 Normalization

We use two normalization procedures in sequence. Dataset-level standardization comes first. Instance normalization follows. These procedures serve different purposes. They are applied at different stages.

First, the dataset-level standardizer is fitted using the training split only. For each variable c, let m_c^{train} and s_c^{train} denote the mean and standard deviation estimated from the training observations. The same statistics are then fixed and applied to the training, validation, and test splits:

$$X_{b,t,c}^{\text{std}} = \frac{X_{b,t,c}^{\text{raw}} - m_c^{\text{train}}}{s_c^{\text{train}}}, \tag{13}$$

where $X_{b,t,c}^{\text{raw}}$ denotes an observation in the original physical scale and $X_{b,t,c}^{\text{std}}$ denotes the corresponding dataset-standardized value. No statistics are estimated from validation or test targets.

After dataset-level standardization, the model applies instance normalization independently to each input sample and each variable. For the standardized input window X^{std}, the mean and variance of variable c in sample b are computed along the temporal dimension:

$$\mu_{b,c} = \frac{1}{L}\sum_{t=1}^{L} X_{b,t,c}^{\text{std}}, \qquad \sigma_{b,c}^2 = \frac{1}{L}\sum_{t=1}^{L}\left(X_{b,t,c}^{\text{std}} - \mu_{b,c}\right)^2. \tag{14}$$

The instance-normalized input is then defined as

$$\widetilde{X}_{b,t,c} = \frac{X_{b,t,c}^{\text{std}} - \mu_{b,c}}{\sqrt{\sigma_{b,c}^2 + \epsilon}}, \tag{15}$$

where $\epsilon = 10^{-5}$ is a small constant used for numerical stability.

Let $\widehat{Y}^{\text{inst}}$ denote the model output in the instance-normalized space. Before evaluation, the model first performs instance-level denormalization using the statistics computed from the corresponding historical input window:

$$\widehat{Y}_{b,t,c}^{\text{std}} = \widehat{Y}_{b,t,c}^{\text{inst}}\sqrt{\sigma_{b,c}^2 + \epsilon} + \mu_{b,c}. \tag{16}$$

The resulting prediction is still in the dataset-standardized space. Therefore, a second inverse transformation is applied using the training-set statistics:

$$\widehat{Y}_{b,t,c}^{\text{raw}} = \widehat{Y}_{b,t,c}^{\text{std}} s_c^{\text{train}} + m_c^{\text{train}}. \tag{17}$$

The final prediction $\widehat{Y}^{\text{raw}}$ is therefore expressed in the original physical scale and is used for reporting forecasting metrics and evaluating physics-based constraints.

A.4 Evaluation Metrics

We report mean squared error and mean absolute error:

$$\mathrm{MSE} = \frac{1}{BHC} \sum_{b=1}^{B} \sum_{t=1}^{H} \sum_{c=1}^{C} \left(\widehat{Y}_{b,t,c} - Y_{b,t,c} \right)^2, \tag{18}$$

and

$$\mathrm{MAE} = \frac{1}{BHC} \sum_{b=1}^{B} \sum_{t=1}^{H} \sum_{c=1}^{C} \left| \widehat{Y}_{b,t,c} - Y_{b,t,c} \right|. \tag{19}$$

Lower MSE and MAE indicate better forecasting performance. For Boiler Equipment, we additionally report physical-consistency metrics described in Section E.

B Implementation Details

B.1 Model Configuration

The implementation uses a temporal state-space encoder followed by variate-token attention. The state-space encoder processes each variable independently with shared parameters. The final state of each variable is then used as its variate token.

The backbone hyperparameters are shared across datasets. Dataset-specific settings are limited to the number of input/output variables, sampling frequency, target column, and the optional physics-profile weights. The optimizer uses Adam without weight decay or explicit gradient clipping. The learning rate follows the Type-1 schedule implemented in the training script, namely 10^{-4} at the first epoch and a factor-of-two reduction after each subsequent epoch. Early stopping monitors validation MSE and restores the best validation checkpoint. All reported runs use the fixed random seed 2023; multi-seed means and standard deviations are therefore not claimed.

B.2 Optimization and Model Selection

The forecasting objective is based on the prediction error. When industrial-prior constraints are enabled, the total training objective is written as

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{pred}} + \lambda_{\mathrm{range}} \mathcal{R}_{\mathrm{range}} + \lambda_{\mathrm{smooth}} \mathcal{R}_{\mathrm{smooth}} + \lambda_{\mathrm{dist}} \mathcal{R}_{\mathrm{dist}} + \lambda_{\mathrm{lag}} \mathcal{R}_{\mathrm{lag}}. \tag{20}$$

The prediction loss is computed from the forecasting output, while the four regularization terms encode operating range, variation magnitude, distribution consistency, and lagged-variable relations. The constraint weights are selected using training and validation information only. After the final configuration is selected, the test set is evaluated once.

B.3 Baseline Implementations

We compare SSVT with TimeEmb (Xia et al., 2025), Mamba3 (Lahoti et al., 2026), Fredformer (Piao et al., 2024), iTransformer (Liu et al., 2024b), and PatchTST (Nie et al., 2023). These baselines represent temporal-embedding models, state-space models, frequency-domain models, variate-token attention models, and patch-based Transformer models, respectively.

To distinguish controlled experiments from literature-level comparisons, we separate locally reproduced baselines from externally reported reference results. For the baselines that were executed locally, the same chronological data partitions, multivariate forecasting setting, input length $L = 96$, forecasting horizons $H \in \{96, 192, 336, 720\}$, evaluation metrics, and fixed random seed are used as for SSVT. The public implementations are adapted only through the data-loading and model-interface layers required by the unified forecasting framework; the original model mechanisms are otherwise preserved.

Table 4: Implementation configuration of SSVT.

Configuration	Value
Input length L	96
Forecasting horizons H	$96, 192, 336, 720$
Token dimension D	64
Feed-forward dimension d_{ff}	128
Number of attention heads	4
Dropout rate	0.1
Activation function	GELU (VTA feed-forward block)
Optimizer	Adam
Initial learning rate	1×10^{-4}
Learning-rate schedule	Type-1: halve after each epoch
Batch size	4
Maximum training epochs	10
Early-stopping patience	3 epochs
Weight decay	0
Gradient clipping	Not used
Random seeds	2023 (one fixed seed)

When a baseline result is taken from the corresponding paper or official result table rather than re-run locally, we align the dataset name, forecasting horizon, multivariate setting, and metric definitions as far as the original report permits. Such values are treated as reported reference results rather than exact reimplementations. In particular, we do not claim that literature-reported values use the same hardware, software version, optimizer settings, training budget, or checkpoint selection procedure as SSVT. The main comparison should therefore be interpreted as a comparison with published benchmark references, whereas the controlled architectural conclusions are obtained from the locally executed ablation experiments.

The local ablation experiments use the same training and evaluation protocol for all variants. Specifically, the complete SSVT model is compared with the Mamba3-only model, the Mamba3 model with ordinary temporal self-attention, and the inverted variate-token attention-only model. These experiments isolate the contribution of the temporal state-space encoder and the inverted variate-token attention module under the same data split, input length, prediction horizon, optimizer, training budget, and random seed.

For the physics-guided experiments, the SSVT backbone is kept fixed and only the auxiliary constraint terms and their weights are changed. The LLM is used only in an outer-loop configuration stage to propose candidate constraint weights. It does not participate in the forward pass, gradient computation, or test-set prediction. All final test results are obtained only after the model and constraint configuration have been selected using the training and validation portions.

B.4 Hardware and Reproducibility

All locally executed experiments are performed in a fixed WSL2 environment on a Windows host using a single NVIDIA GeForce RTX 5060 Laptop GPU with approximately 8 GB of VRAM. The recorded GPU driver version is 582.05, and the CUDA runtime used by PyTorch is CUDA 12.8. The experiments are executed in the Linux mamba3 environment with Python 3.10, PyTorch 2.11.0+cu128, and mamba-ssm version 2.3.2.post1.

The main SSVT configuration uses a single GPU, data-loader worker count zero, disabled automatic mixed precision, and the fixed random seed 2023. The remaining model and optimization settings are listed in Table 4. The CPU model and host memory size were not recorded in the original experiment metadata; therefore, no claim about CPU-dependent performance or system-memory requirements is made.

A separate inference benchmark was performed for the 720-step forecasting setting using batch size one, 20 warm-up iterations, and 100 measured iterations. On ETTh1, SSVT

contains 130,836 trainable parameters and requires approximately 1.538 ms per inference batch, corresponding to approximately 650.4 samples per second. On Boiler Equipment, the same model contains 130,836 trainable parameters and requires approximately 1.810 ms per inference batch, corresponding to approximately 552.4 samples per second. These measurements describe inference throughput only; they should not be interpreted as average end-to-end training time.

C Additional Model Details

C.1 Temporal State-Space Encoder

Let

$$\widetilde{X} \in \mathbb{R}^{B\times L\times C}$$

denote the normalized input window, where B is the batch size, L is the historical input length, and C is the number of variables. The scalar input of variable c at time step t for sample b is denoted by $\widetilde{x}_{b,t,c}$.

Each variable is first embedded independently into the token space:

$$u^{(0)}_{b,t,c} = W_{\text{in}}\widetilde{x}_{b,t,c} + b_{\text{in}} + p_t, \tag{21}$$

where $W_{\text{in}} \in \mathbb{R}^{D\times 1}$ and $b_{\text{in}} \in \mathbb{R}^D$ are learnable input-embedding parameters, $p_t \in \mathbb{R}^D$ is a learnable temporal-position embedding, and D is the token dimension. Thus, $u^{(0)}_{b,t,c} \in \mathbb{R}^D$.

The temporal encoder processes each variable along the time axis using shared Mamba3 parameters. Let N_t denote the number of TSSE layers. For the ℓ-th TSSE layer, the temporal sequence of variable c is updated as

$$r^{(\ell)}_{b,c} = \text{Mamba3}_{\theta_\ell}\left(u^{(\ell-1)}_{b,c}\right), \quad \ell = 1,\dots,N_t, \quad r^{(\ell)}_{b,c} \in \mathbb{R}^L. \tag{22}$$

where θ_ℓ denotes the trainable parameters of the ℓ-th Mamba3 layer and $r^{(\ell)}_{b,t,c}$ is its output at time step t.

The residual connection and layer normalization used in the implementation can be written as

$$u^{(\ell)}_{b,t,c} = \text{LN}\left(u^{(\ell-1)}_{b,t,c} + \text{Dropout}\left(r^{(\ell)}_{b,t,c}\right)\right), \tag{23}$$

where LN denotes layer normalization and Dropout denotes the dropout operation.

At an abstract level, the Mamba3 operation can be viewed as a selective state-space recurrence:

$$s^{(\ell)}_{b,t,c} = \overline{A}^{(\ell)}_t s^{(\ell)}_{b,t-1,c} + \overline{B}^{(\ell)}_t u^{(\ell-1)}_{b,t,c}, \tag{24}$$

$$r^{(\ell)}_{b,t,c} = C^{(\ell)}_t s^{(\ell)}_{b,t,c} + D^{(\ell)}_t u^{(\ell-1)}_{b,t,c}, \tag{25}$$

where $s^{(\ell)}_{b,t,c}$ is the hidden state of the state-space recurrence, $\overline{A}^{(\ell)}_t$ and $\overline{B}^{(\ell)}_t$ are discretized transition and input operators, and $C^{(\ell)}_t$ and $D^{(\ell)}_t$ are output and skip operators. In the selective state-space implementation, these operators may depend on the current input and time step. The above equations describe the state-space mechanism at a conceptual level and do not replace the internal Mamba3 parameterization.

After the final TSSE layer, the hidden representation at the last historical time step is selected as the token of variable c:

$$z_{b,c} = u^{(N_t)}_{b,L,c} \in \mathbb{R}^D. \tag{26}$$

Therefore, the temporal encoder transforms the input tensor from $[B, L, C]$ into a set of variable tokens:

$$Z = [z_{b,1}, z_{b,2}, \dots, z_{b,C}] \in \mathbb{R}^{B\times C\times D}. \tag{27}$$

Each token represents one sensor variable, while its feature values summarize the complete historical trajectory of that variable. The temporal encoder therefore models within-variable temporal evolution before cross-variable interaction is performed.

C.2 Variate-Token Attention

Given

$$Z \in \mathbb{R}^{B\times C\times D},$$

the variate-token attention module performs self-attention along the variable dimension. In other words, the C sensor variables are treated as tokens, while the temporal dimension has already been compressed by the TSSE.

For attention head m, the query, key, and value representations are computed as

$$Q^{(m)} = ZW_Q^{(m)}, \qquad K^{(m)} = ZW_K^{(m)}, \qquad V^{(m)} = ZW_V^{(m)}, \tag{28}$$

where $W_Q^{(m)}$, $W_K^{(m)}$, and $W_V^{(m)}$ are learnable projection matrices, and m indexes the attention head. If the number of attention heads is N_h, the dimension of each head is $d_h = D/N_h$.

The attention matrix for head m is

$$A^{(m)} = \mathrm{softmax}_{\mathrm{var}}\left(\frac{Q^{(m)}(K^{(m)})^{\mathsf{T}}}{\sqrt{d_h}}\right), \tag{29}$$

where $(K^{(m)})^{\mathsf{T}}$ denotes transposition over the variable token dimension, d_h is the feature dimension of one attention head, and $\mathrm{softmax}_{\mathrm{var}}$ normalizes the attention scores over the C variable tokens.

The output of one attention head is

$$O^{(m)} = A^{(m)}V^{(m)}. \tag{30}$$

The outputs of all attention heads are concatenated and projected back to the token dimension:

$$\mathrm{MHA}(Z) = \mathrm{Concat}\left(O^{(1)}, \ldots, O^{(N_h)}\right) W_O, \tag{31}$$

where W_O is the output projection matrix and $\mathrm{MHA}(Z)$ has shape $\mathbb{R}^{B\times C\times D}$.

The implemented attention block contains residual connections, layer normalization, and a feed-forward network:

$$\overline{Z} = \mathrm{LN}\left(Z + \mathrm{Dropout}\left(\mathrm{MHA}(Z)\right)\right), \tag{32}$$

$$Z' = \mathrm{LN}\left(\overline{Z} + \mathrm{Dropout}\left(\mathrm{FFN}(\overline{Z})\right)\right), \tag{33}$$

where $\overline{Z}$ is the intermediate token representation and Z' is the final representation after variate-level interaction.

The feed-forward network is defined as

$$\mathrm{FFN}(U) = W_2\,\mathrm{GELU}\left(W_1U + b_1\right) + b_2, \tag{34}$$

where W_1 and W_2 are learnable linear projection matrices, b_1 and b_2 are bias vectors, and GELU is the Gaussian error linear unit activation function. The intermediate feed-forward dimension is denoted by d_{ff}.

The attention operation is performed only over the C variable tokens. It does not apply a global self-attention operation over all $L\times C$ time-variable positions. Consequently, temporal modeling and cross-variable modeling are explicitly separated.

C.3 Forecasting Head

The final token of variable c is mapped to its future trajectory by a linear prediction head:

$$\widehat{Y}_{b,c} = W_{\mathrm{out}}z'_{b,c} + b_{\mathrm{out}}, \quad \widehat{Y}_{b,c} \in \mathbb{R}^H, \tag{35}$$

where $W_{\mathrm{out}} \in \mathbb{R}^{H\times D}$ is the output projection matrix, $b_{\mathrm{out}} \in \mathbb{R}^H$ is the output bias, and H is the forecasting horizon.

Table 5: Effect of industrial-prior soft constraints on Boiler Equipment. All entries are single-run test-set point estimates reported.

Horizon	Configuration	MSE	MAE	Range violation	Smoothness violation	JS divergence	Lag violation
96	No constraint	0.102443	0.182685	0.048288	0.077110	0.422134	0.501573
96	v2_pct_lag	0.098365	0.169019	0.047857	0.076977	0.420247	0.494493
720	No constraint	0.648427	0.608321	0.022669	0.071542	0.417938	0.502217
720	Transferred v2_pct_lag	0.640145	0.600868	0.023328	0.094484	0.419361	0.500381

The prediction head is applied to every variable token independently:

$$\widehat{Y} \in \mathbb{R}^{B\times H\times C}. \tag{36}$$

Although the output projection is applied independently to each token, each token already contains information from other variables through the variate-token attention module. Therefore, the final forecast for variable c preserves its variable correspondence while incorporating cross-variable information.

When instance normalization is enabled, the normalized prediction is mapped back to the original physical scale using the statistics of the corresponding input window:

$$\widehat{y}^{\text{raw}}_{b,t,c} = \widehat{y}^{\text{norm}}_{b,t,c}\sigma_{b,c} + \mu_{b,c}, \tag{37}$$

where $\mu_{b,c}$ and $\sigma_{b,c}$ are the mean and standard deviation computed from the historical window of variable c for sample b.

C.4 Computational Complexity

Let D denote the token dimension, d_{ff} the feed-forward dimension, L the historical input length, C the number of variables, H the forecasting horizon, and B the batch size. Ignoring the implementation-dependent state dimension and constant projection factors, the temporal state-space encoder has approximately linear complexity in the historical length, given by $\mathcal{O}(BCLD)$. The variate-token attention module incurs approximately $\mathcal{O}(BC^2D)$ attention complexity and $\mathcal{O}(BCDd_{\text{ff}})$ feed-forward complexity, while the forecasting head requires approximately $\mathcal{O}(BCHD)$ operations to generate the future trajectories. Therefore, the dominant temporal cost of SSVT grows linearly with L. In contrast to a full temporal self-attention layer, which would introduce a quadratic term of $\mathcal{O}(BCL^2D)$, SSVT first performs temporal state-space scanning and then applies self-attention only after the historical sequence has been compressed into C variable tokens. This factorized design avoids direct global attention over all temporal positions and variables.

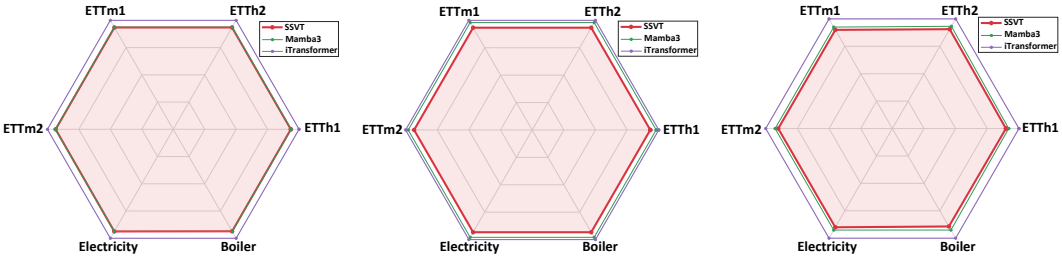
D Additional Experimental Results

D.1 Main Forecasting Results

The complete four-horizon forecasting table is reported in the main paper. The supplied main-results workbook confirms that SSVT obtains the best MSE and MAE at the 720-step horizon on all six datasets (ETTh1, ETTh2, ETTm1, ETTm2, Electricity, and Boiler Equipment). On Boiler Equipment, SSVT obtains an MSE of 0.648427 and an MAE of 0.608321 at this horizon. When averaged over the four horizons on Boiler Equipment, the strongest baseline is TimeEmb. Its average MSE and MAE are 0.424750 and 0.451750, respectively, compared with 0.395234 and 0.427665 for SSVT. These are relative reductions of 6.95% in MSE and 5.33% in MAE.

D.2 Physical-Consistency Results

Table 5 reports the physical-guided results available. The 96-step constrained model uses v2_pct_lag; the same weights are transferred to the 720-step setting.



(a) Maximum GPU memory (b) Number of parameters (c) Inference latency

Figure 5: Efficiency comparison of SSVT, Mamba3, and iTransformer on six forecasting datasets.

At the 96-step horizon, the constrained model reduces MSE from 0.102443 to 0.098365 (3.98%). It reduces MAE from 0.182685 to 0.169019 (7.48%). All four reported physical metrics also decrease. Range violation changes from 0.048288 to 0.047857. Smoothness violation changes from 0.077110 to 0.076977. JS divergence changes from 0.422134 to 0.420247. Lag violation changes from 0.501573 to 0.494493. At the 720-step horizon, the transferred configuration reduces MSE from 0.648427 to 0.640145 (1.28%). It reduces MAE from 0.608321 to 0.600868 (1.23%). The physical metrics do not all move in the same direction. Range violation increases from 0.022669 to 0.023328. Smoothness violation increases from 0.071542 to 0.094484. JS divergence increases from 0.417938 to 0.419361. Lag violation decreases from 0.502217 to 0.500381.

D.3 Efficiency Comparison

SSVT first uses the Mamba3-based Temporal State-Space Encoder (TSSE) to model the temporal evolution of each variable independently. It then treats the final hidden state of each variable as a variate token, while the Inverted Variate-Token Attention (IVTA) module enables interactions across variables. Thus, SSVT jointly performs temporal state encoding and cross-variable interaction. As shown in Fig. 5, SSVT is not the most computationally efficient across all metrics. Relative to Mamba3, it adds a variate-token attention block, feed-forward network, residual connections, and normalization layers, increasing the parameter count and inference latency. Relative to iTransformer, SSVT performs Mamba3-based temporal encoding before variate interaction, whereas iTransformer directly maps each historical variable sequence to a variate token. Consequently, iTransformer may be more efficient for shorter input windows. This trade-off reflects the balance between efficiency and representation capacity. SSVT incurs additional computation to capture both individual temporal dynamics and cross-variable dependencies. Its main advantage is therefore the joint modeling of temporal evolution and cross-variable relationships, rather than minimum latency or model size. This overhead is acceptable in the intended industrial forecasting setting when gains in forecasting accuracy and physical consistency justify the additional computation.

D.4 Ablation Results

We analyze the contribution of TSSE. We analyze the contribution of VTA separately. We use four model variants. The first is w/o VTA. The second is w/ Temporal Attention. The third is w/o TSSE. The fourth is Full SSVT. All variants use the same data partitions and the 720-step forecasting horizon.

Full SSVT obtains the lowest MSE and MAE for both datasets in the supplied ablation table. On ETTh1, relative to w/o VTA, it reduces MSE by 16.15% and MAE by 11.52%; relative to w/o TSSE, the reductions are 7.11% and 6.40%, respectively. On Boiler Equipment, the corresponding reductions relative to w/o VTA are 16.92% in MSE and 4.85% in MAE, and relative to w/o TSSE are 16.82% and 4.85%. The temporal-attention variant (TSSE + temporal attention) does not match Full SSVT: on ETTh1 it is worse than TSSE-only

Table 6: Ablation results for the hybrid backbone at the 720-step forecasting horizon.

Dataset	Variant	MSE	MAE
ETTh1	w/o VTA	0.545	0.512
ETTh1	w/ Temporal Attention	0.566	0.528
ETTh1	w/o TSSE	0.492	0.484
ETTh1	Full SSVT	0.457	0.453
Boiler Equipment	w/o VTA	0.780	0.639
Boiler Equipment	w/ Temporal Attention	0.779	0.638
Boiler Equipment	w/o TSSE	0.779	0.639
Boiler Equipment	Full SSVT	0.648	0.608

(MSE/MAE 0.566/0.528 versus 0.545/0.512), while on Boiler Equipment it is nearly tied with TSSE-only (0.779/0.638 versus 0.780/0.639).

D.5 Visualization of Prediction Results

To complement the numerical results, we visualize representative SSVT forecasts at the 96-step and 192-step horizons. Figure 6 shows the first three datasets, namely ETTh1, ETTh2, and Electricity. For each dataset, the 96-step and 192-step forecasts are placed in the same row, and each panel overlays the ground-truth sequence with the corresponding SSVT prediction. These panels provide a qualitative view of the forecast trajectories without introducing predictions from other models.

Figure 7 presents the same visualization for the remaining three datasets: ETTm1, ETTm2, and Boiler Equipment. The same layout is used so that the two forecast horizons can be inspected within each dataset. As in Figure 6 and Figure 7, these plots are qualitative illustrations of SSVT predictions rather than a comparison against other forecasting models.

E Industrial-Prior Constraint Details

Let $\widehat{Y}^{\text{phys}}$ denote the model prediction after inverse scaling, namely in the original physical units. Here, B is the batch size, H is the forecasting horizon, and C is the number of variables.

E.1 Operating-Range Constraint

For Boiler Equipment, the operating-range prior is applied to the boiler outlet temperature rather than to all sensor variables. Let $\widehat{y}^{T}_{b,t}$ denote the predicted outlet temperature for sample b and future step t. The accepted operating interval is $[\ell_T, u_T] = [530, 545]$ degrees Celsius. The soft range penalty is defined as

$$\mathcal{R}_{\text{range}} = \frac{1}{BH} \sum_{b=1}^{B} \sum_{t=1}^{H} \left[\text{ReLU}\left(\frac{\ell_T - \widehat{y}^{T}_{b,t}}{s_T} \right)^2 + \text{ReLU}\left(\frac{\widehat{y}^{T}_{b,t} - u_T}{s_T} \right)^2 \right], \tag{38}$$

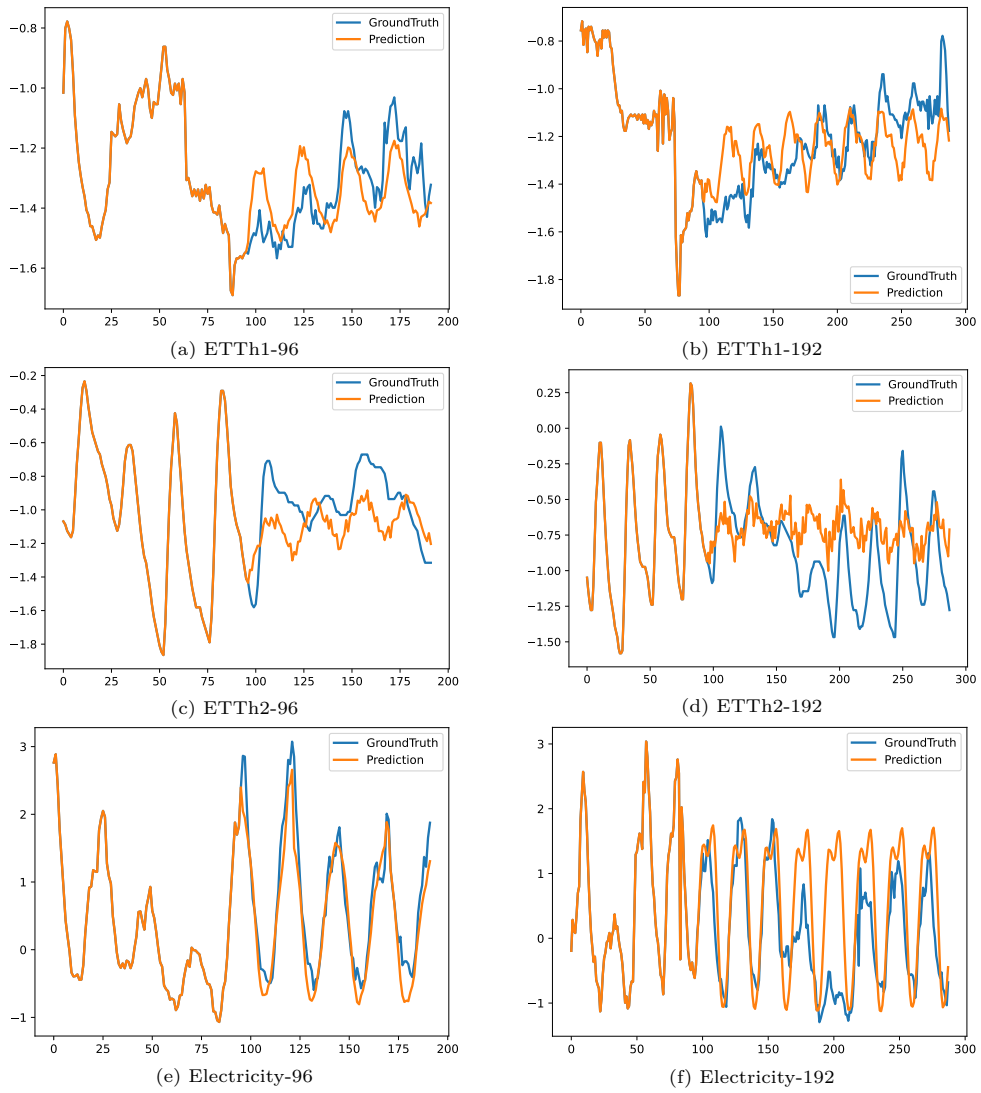
where $\text{ReLU}(x) = \max(0, x)$, and s_T is the training-set standard deviation of the outlet-temperature variable used to keep the penalty numerically comparable with the forecasting loss.

The interval $[530, 545]$ is treated as a normal-operation boundary identified for the Boiler Equipment data. It is therefore a soft operating prior rather than a strict physical law. The constraint is enabled only for the Boiler Equipment profile and is not applied to variables whose engineering limits have not been established.

E.2 Variation-Magnitude Constraint

Let $\widehat{y}^{c}_{b,0}$ denote the last observed value in the input window and let $\widehat{y}^{c}_{b,t}$ denote the predicted value at future step t. The one-step predicted variation is

$$\Delta\widehat{y}^{c}_{b,t} = \widehat{y}^{c}_{b,t} - \widehat{y}^{c}_{b,t-1}, \qquad t = 1, \ldots, H. \tag{39}$$



(a) ETTh1-96 (b) ETTh1-192

(c) ETTh2-96 (d) ETTh2-192

(e) Electricity-96 (f) Electricity-192

Figure 6: Qualitative visualization of SSVT forecasts on ETTh1, ETTh2, and Electricity. Each row corresponds to one dataset, with the 96-step and 192-step forecasts shown from left to right.

A training-set threshold τ_c is estimated as the 95th percentile of the absolute one-step variation:

$$\tau_c = \text{Quantile}_{0.95}\left(|\Delta y_t^c|\right), \tag{40}$$

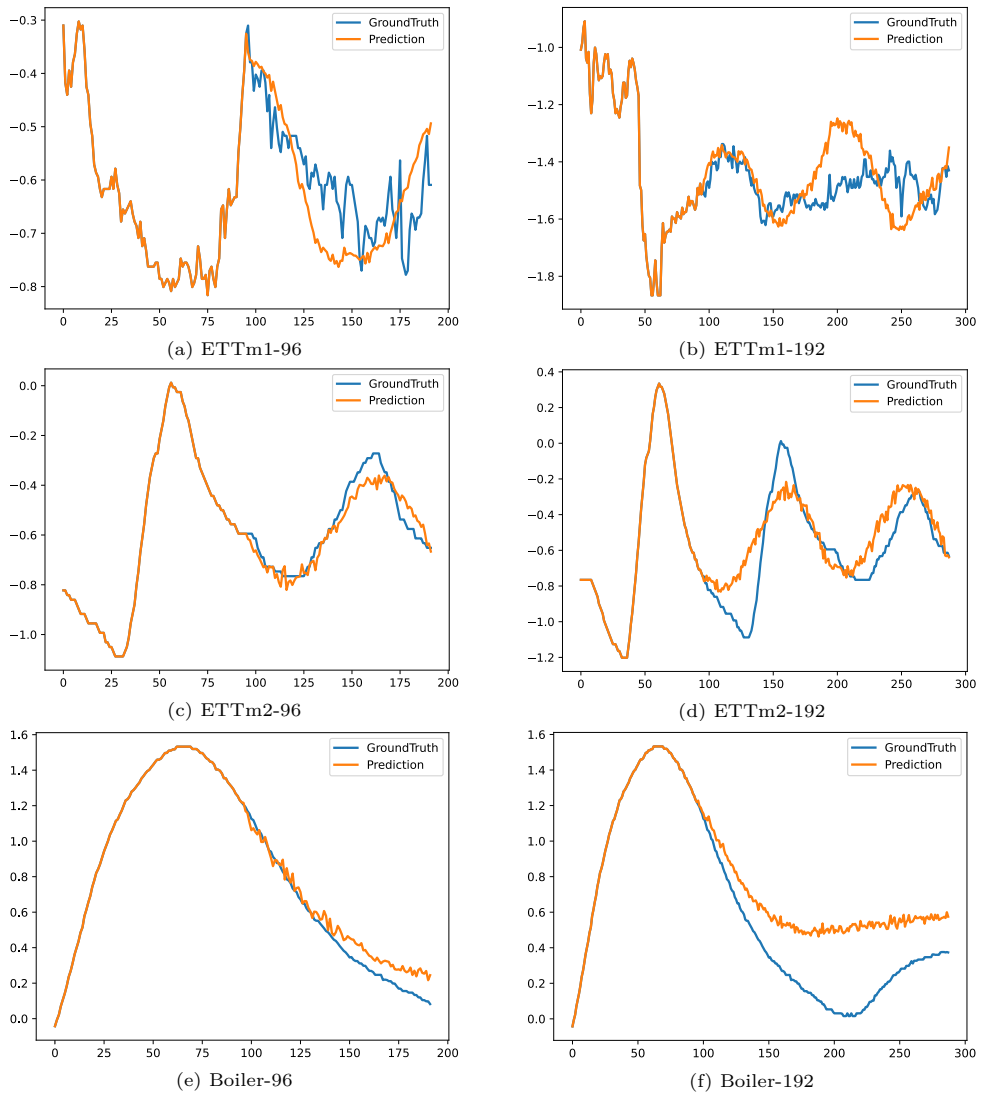
where y_t^c denotes the observed training-set value of variable c. The threshold is calculated in the original physical units using training observations only.

The variation penalty is

$$\mathcal{R}_{\text{smooth}} = \frac{1}{BHC} \sum_{b,t,c} \text{ReLU}\left(\frac{\left|\Delta \widehat{y}_{b,t}^c\right| - \tau_c}{s_c} \right)^2, \tag{41}$$

where s_c is the training-set standard deviation of variable c.

For Boiler Equipment, this constraint is applied to the outlet temperature. For the ETT profile, it is applied to the oil-temperature variable OT. The same training-derived threshold is used for all forecasting horizons of the corresponding dataset. The penalty only discourages unusually large local changes and does not force the whole forecast to be constant.



(a) ETTm1-96 (b) ETTm1-192

(c) ETTm2-96 (d) ETTm2-192

(e) Boiler-96 (f) Boiler-192

Figure 7: Qualitative visualization of SSVT forecasts on ETTm1, ETTm2, and Boiler Equipment. Each row corresponds to one dataset, with the 96-step and 192-step forecasts shown from left to right.

For the cleaned Boiler Equipment data, the outlet-temperature threshold used in the experiments is approximately 0.1300 degrees Celsius per sampling step.

E.3 Distribution-Consistency Constraint

The distribution prior is used for the Boiler outlet temperature. Let P_T denote the reference distribution estimated from the training portion and let $\widehat{P}_T$ denote the distribution of the predicted temperature values. To obtain a differentiable approximation, both distributions are represented by a soft histogram with $K = 32$ fixed centers $\{\mu_k\}_{k=1}^{K}$ over the interval $[530, 545]$.

For a set of temperature values $\{v_i\}$, its soft histogram is defined as

$$p_k = \frac{\sum_i \exp\left(-\frac{(v_i-\mu_k)^2}{2\sigma^2}\right)}{\sum_{j=1}^{K}\sum_i \exp\left(-\frac{(v_i-\mu_j)^2}{2\sigma^2}\right)}, \tag{42}$$

where μ_k is the k-th histogram center and σ controls the smoothness of the histogram. The reference probabilities $P_T(k)$ are computed from training observations, while the predicted probabilities $\widehat{P}_T(k)$ are computed from the forecast.

The distribution penalty is the Jensen–Shannon divergence:

$$\mathcal{R}_{\text{dist}} = \text{JS}\left(P_T, \widehat{P}_T\right), \tag{43}$$

where

$$\text{JS}(P, Q) = \frac{1}{2}\,\text{KL}(P\|M) + \frac{1}{2}\,\text{KL}(Q\|M), \qquad M = \frac{1}{2}(P + Q). \tag{44}$$

Here, KL denotes the Kullback–Leibler divergence and M is the average distribution of the reference and predicted probabilities. The reference distribution, histogram centers, and smoothing scale are obtained without using test observations.

E.4 Lag-Relation Constraint

The Boiler analysis considers the empirical relation between cooling-water flow and outlet-temperature changes. Let q_t denote the cooling-water flow variable and let y_t denote the boiler outlet temperature. In the current implementation, the cooling-water variable is YJJWSLL.AV_0#, the temperature variable is TE_8332A.AV_0#, and the lag is $\ell = 60$ sampling steps. Because the Boiler data are sampled every 5 seconds, this corresponds to approximately 5 minutes.

Define

$$\Delta q_t = q_{t+1} - q_t, \qquad \Delta y_{t+\ell} = y_{t+\ell+1} - y_{t+\ell}. \tag{45}$$

The expected relation is a weak negative association: an increase in cooling-water flow should not be systematically accompanied by an increase in outlet temperature at the selected lag. The corresponding soft penalty is

$$\mathcal{R}_{\text{lag}} = \frac{1}{N_\ell} \sum_t \text{ReLU}\left(\frac{\Delta q_t \Delta y_{t+\ell}}{s_q s_y}\right)^2, \tag{46}$$

where s_q and s_y are the training-set standard deviations of the cooling-water flow and outlet temperature, respectively, and N_ℓ is the number of valid lagged pairs.

The penalty is positive when the two changes have the same sign and is zero when they have opposite signs. The relation is an empirical correlation prior and should not be interpreted as a causal law.

For evaluation, the lag-relation violation rate is calculated as

$$V_{\text{lag}} = \frac{\sum_t \mathbb{I}\left(\Delta q_t \Delta y_{t+\ell} > 0\right)}{\sum_t \mathbb{I}\left(\Delta q_t \neq 0 \ \wedge \ \Delta y_{t+\ell} \neq 0\right)}, \tag{47}$$

where $\mathbb{I}(\cdot)$ is the indicator function. Zero changes are excluded because they provide no directional evidence.

E.5 Interpretation of the Constraints

The four terms are auxiliary soft regularizers. They do not constitute a strict physical simulator and do not solve a governing differential equation. Their purpose is to discourage predictions that violate observed operating ranges, training-set variation magnitudes, historical temperature distributions, or empirically observed lagged relations.

The constraint weights determine the balance between forecasting accuracy and prior consistency:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{pred}} + \lambda_{\text{range}}\mathcal{R}_{\text{range}} + \lambda_{\text{smooth}}\mathcal{R}_{\text{smooth}} + \lambda_{\text{dist}}\mathcal{R}_{\text{dist}} + \lambda_{\text{lag}}\mathcal{R}_{\text{lag}}, \tag{48}$$

where $\mathcal{L}_{\text{pred}}$ is the supervised MSE loss and the four λ values are non-negative constraint weights. When a constraint is not used, its corresponding weight is set to zero.

Because these priors are approximate, their effects may vary with the dataset, operating condition, and forecasting horizon. Therefore, both forecasting metrics and physical-consistency metrics are reported.

F LLM-Assisted Weight Configuration

F.1 Role of the LLM

The LLM is used as an outer-loop assistant for selecting the weights of the industrial-prior constraints. Let

$$\boldsymbol{\lambda} = (\lambda_{\text{range}}, \lambda_{\text{smooth}}, \lambda_{\text{dist}}, \lambda_{\text{lag}})$$

denote the four constraint weights.

The LLM does not modify the SSVT architecture and does not participate in the forward pass, gradient computation, parameter update, or time-series prediction. Its only role is to analyze the training diagnostics and validation results and propose a candidate $\boldsymbol{\lambda}$ for the next training round.

F.2 Candidate Generation

For the current configuration r, the diagnostic information is summarized as

$$\mathcal{D}^{(r)} = \left[\text{MSE}_{\text{val}}^{(r)}, \text{MAE}_{\text{val}}^{(r)}, V_{\text{range}}^{(r)}, V_{\text{smooth}}^{(r)}, J_{\text{dist}}^{(r)}, V_{\text{lag}}^{(r)}, \boldsymbol{\lambda}^{(r)}\right], \tag{49}$$

where MSE_{val} and MAE_{val} are validation errors, V_{range} and V_{smooth} are physical violation rates, J_{dist} is the distribution discrepancy, and V_{lag} is the lag-relation violation rate.

The LLM receives these diagnostics together with the meanings of the constraints and the valid search ranges. It then proposes

$$\widetilde{\boldsymbol{\lambda}}^{(r+1)} = \text{LLM}\left(\mathcal{D}^{(r)}, \mathcal{R}_{\text{phys}}, \mathcal{B}_{\lambda}\right), \tag{50}$$

where $\mathcal{R}_{\text{phys}}$ denotes the available industrial-prior descriptions and $\mathcal{B}_{\lambda}$ denotes the allowed search region.

The weights are restricted to

$$\begin{aligned} &0 \le \lambda_{\text{range}} \le 0.003, \\ &0 \le \lambda_{\text{smooth}} \le 0.003, \\ &0 \le \lambda_{\text{dist}} \le 0.08, \\ &0 \le \lambda_{\text{lag}} \le 0.003. \end{aligned} \tag{51}$$

F.3 Numerical Guardrails and Selection

The LLM output is checked before it is used:

$$\boldsymbol{\lambda}^{(r+1)} = \text{Clip}_{\mathcal{B}_{\lambda}}\left[\text{Validate}\left(\widetilde{\boldsymbol{\lambda}}^{(r+1)}\right)\right]. \tag{52}$$

Here, $\text{Validate}(\cdot)$ checks whether all required weights are present and numerically valid, while $\text{Clip}_{\mathcal{B}_{\lambda}}(\cdot)$ restricts the candidate to the predefined search region.

The selection procedure is:

1. Train the unconstrained SSVT baseline.
2. Evaluate a small number of manually selected constraint configurations on the validation set.
3. Provide the validation diagnostics and constraint definitions to the LLM.
4. Generate one bounded neighboring configuration.
5. Train and evaluate the proposed configuration on the validation set.

6. Select the final configuration according to validation MSE, with physical-consistency metrics used as secondary evidence.
7. Freeze the selected configuration before the final test evaluation.

The manual search contains five candidate configurations, while the LLM-assisted procedure proposes one additional candidate. The LLM is therefore used to reduce unnecessary trial-and-error rather than to perform a large unrestricted hyperparameter search.

The selected configuration can be written as

$$r^* = \arg\min_{r \in \mathcal{V}} \mathrm{MSE}_{\mathrm{val}}^{(r)}, \tag{53}$$

where $\mathcal{V}$ is the set of valid candidate configurations evaluated on the validation set.

F.4 Interpretation

The LLM is a configuration assistant rather than a forecasting model. Its contribution is evaluated by the number of candidate trials, the validity of its proposed configuration, and the balance between forecasting accuracy and physical consistency. It does not directly produce predictions and is not required during inference.

F.5 AI-Assisted Configuration Statement

The LLM was used only to propose weights for the industrial-prior regularizers in an outer-loop validation procedure. It was not used in the SSVT forward computation, gradient calculation, parameter optimization, or time-series prediction. After validation-based model selection, the chosen configuration was fixed before the final test evaluation.