

QuadFocusNet: Learning to Route Multi-Scale Patches for Multivariate Time Series Forecasting

Jiayang Xu

Department of Statistics and Data Science
Southern University of Science and Technology
Shenzhen, China
12313323@mail.sustech.edu.cn

Abstract—Time series forecasting is essential yet challenging across various domains such as energy management, climate monitoring, and traffic flow prediction. Recent methods, such as PatchTST, have achieved strong performance. However, they rely on fixed-scale patching, which limits adaptability to non-stationary distributions and diverse temporal patterns. Other approaches, such as DLinear, are efficient but struggle to capture complex nonlinear dependencies. To address these limitations, we propose QuadFocusNet, a novel framework built upon multi-scale adaptive patching. Instead of using a static patch size, QuadFocusNet extracts multi-granular temporal representations at multiple scales in parallel. It then dynamically fuses them through a learnable router, enabling content-aware scale selection and efficient information flow. Moreover, we incorporate prompt learning to inject global context and improve representation robustness under distribution shifts. Extensive experiments on the ETT, UCI Household Power Consumption, and Jena Climate datasets demonstrate that QuadFocusNet achieves competitive state-of-the-art performance. The model consistently delivers superior or competitive results across diverse settings. These range from immediate prediction tasks to medium-term forecasting at 24- and 48-step horizons, up to long-term forecasting at a 96-step horizon. Code is available at <https://github.com/David-SUSTech/QuadFocusNet.git>.

Index Terms—Multivariate Time Series Forecasting, Adaptive Patching, Transformer, Prompt Tuning.

I. INTRODUCTION

Multivariate time series forecasting (MTSF) is essential across various domains, including energy management [1]–[3], climate monitoring [4], and traffic flow prediction [5]–[7]. These applications require high levels of accuracy over different time horizons, from real-time one-step-ahead control to long-term strategic planning.

However, maintaining consistent performance across such different horizons remains challenging. Classical statistical models such as ARIMA [8] often perform well in short-term forecasting but struggle in long-horizon settings due to their limited ability to capture complex nonlinear interactions and changing temporal dynamics. Conversely, Transformer-based models [9], [10] have become strong performers on long-term benchmarks but may miss detailed local semantics needed for short-term predictions due to their global attention mechanisms. This performance gap is often linked to an insufficient inductive bias related to local temporal variations.

Existing state-of-the-art models also have structural limitations. PatchTST [11] relies on a fixed patch size, ignoring that

optimal scales vary across instances. This causes a rigid trade-off: large patches may smooth out critical local fluctuations, while small patches may fail to capture global trends and become more vulnerable to noise. TimesNet [12] explicitly models periodicity but can introduce additional computational overhead, while DLinear is efficient but limited in expressing complex nonlinear relationships. Thus, there is an urgent need for a unified framework that can adaptively capture both high-frequency local variations and low-frequency global trends.

To address these challenges, we propose QuadFocusNet, a decomposition-based architecture designed for robust cross-horizon forecasting. Instead of inflexible patching, QuadFocusNet employs a Multi-scale Adaptive Patching approach to extract multi-granular representations with parallel receptive fields. By explicitly restoring local inductive bias through small-scale patches, the model functions as a “variable-focus lens” that simultaneously captures fine-grained details and coarse-grained trends. Crucially, we introduce a Learnable Router that adaptively weights and combines these multi-resolution features based on input characteristics. Additionally, we incorporate Prompt Tuning to enhance generalization and integrate RevIN [14] with Huber loss to improve stability during distribution shifts and outliers.

We evaluate QuadFocusNet on the ETT, UCI-HPC, and Climate datasets, covering both one-step-ahead and 96-step long-horizon forecasting. Experimental results demonstrate strong performance across both settings, confirming the model’s robustness over different horizons. The main contributions are summarized as follows:

- We propose QuadFocusNet, a unified framework integrating series decomposition with Multi-scale Adaptive Patching, effectively alleviating the trade-off between preserving local details and capturing global trends.
- We design a Learnable Router that dynamically combines multi-scale features and adaptively selects informative receptive fields for both short-term and long-term forecasting.
- Extensive evaluations on the ETT, UCI-HPC, and Climate datasets demonstrate that QuadFocusNet consistently delivers strong and superior performance compared to state-of-the-art baselines across various forecasting scenarios.

II. RELATED WORK

a) *Statistical time series modeling*: Classical methods emphasize interpretability and statistical rigor. ARIMA [8] models stationary series using autoregressive and moving-average components, while Vector Autoregression (VAR) [15] expands this approach to capture linear dependencies among multiple variables. Although effective under stable conditions, these models are limited by linear assumptions, which restrict their ability to capture complex nonlinear dynamics and high-dimensional multivariate relationships.

b) *Deep learning-based models*: Early deep learning approaches employed recurrent architectures such as LSTMs and GRUs [16]–[19] to model nonlinear temporal dynamics. Convolutional models, especially Temporal Convolutional Networks (TCNs) [20]–[23], leverage dilated convolutions to effectively widen receptive fields. Despite improved flexibility, recurrent models are more computationally expensive because of their sequential nature, and convolutional backbones often lack an explicit way to model content-dependent global dependencies compared to attention-based architectures.

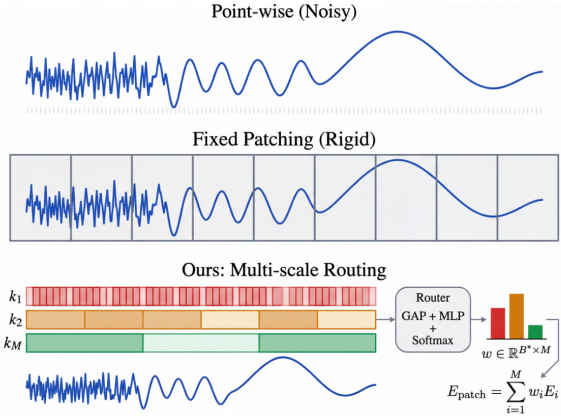


Fig. 1: A schematic comparison of representative time series forecasting paradigms; our approach is shown in the bottom row.

c) *Transformer variants*: Transformers [24]–[26] excel at modeling long-range dependencies but face quadratic complexity challenges. Informer [27] mitigates this with sparse attention mechanisms. Subsequent works incorporate specific inductive biases: Autoformer [28] and FEDformer [29] combine series decomposition with autocorrelation and frequency-domain operations, respectively, to improve efficiency and trend detection.

d) *Recent advances*: Recent approaches emphasize refined representation learning and structural efficiency. For instance, DLinear [13] demonstrates that a simple linear decomposition baseline can be highly effective. Building on this idea, PatchTST [11] introduces patch-based tokenization with channel independence to capture local semantic patterns while reducing computational redundancy. Extending the focus further, TimesNet [12] transforms 1D series into 2D tensors for multi-periodic analysis. In contrast, iTransformer [32] applies

attention across multivariate dimensions rather than time steps to better capture inter-series correlations.

III. METHODOLOGY

A. Problem Definition and Overall Architecture

We study multivariate time series forecasting with an input context window of length L . For each sample, the input is $X \in \mathbb{R}^{L \times C}$, where C denotes the number of channels. The forecasting horizon is T , and the model outputs a T -step prediction for the target variable τ ,

$$\hat{y}_\tau \in \mathbb{R}^T.$$

Our experiments include both *single-step* and *multi-step* settings: we use $T \in \{1, 24, 48\}$ on ETT, $T = 1$ on UCI-HPC, and $T = 96$ on the Climate dataset.

The lookback length L follows dataset protocols. Specifically, $L = 168$ for ETT, and $L = 336$ for UCI-HPC and Climate.

Let the last observed values in the input window be

$$x^{\text{last}} = X_{L,:} \in \mathbb{R}^C, \quad x_\tau^{\text{last}} = X_{L,\tau}. \quad (1)$$

Our final prediction is formed by anchoring on x_τ^{last} and adding a learned residual.

a) *Channel Independence protocol*: We adopt a channel-independent (CI) processing strategy. Given a batch input $\mathcal{X} \in \mathbb{R}^{B \times L \times C}$, we permute and reshape it into

$$\tilde{\mathcal{X}} \in \mathbb{R}^{B^* \times L \times 1},$$

where $B^* = BC$. Each channel is treated as an individual univariate instance, while all channels share the same model weights. After prediction, we reshape the CI outputs back to $\mathbb{R}^{B \times T \times C}$ and select the target channel τ .

b) *Overall architecture*: QuadFocusNet follows a dual-stream design. The input is first normalized using instance-wise, channel-wise normalization, and then a moving-average decomposition produces seasonal and trend components. The trend stream uses a lightweight regressor to model low-frequency variations. The seasonal stream performs adaptive patching to generate tokens and applies a Transformer encoder for representation learning. The two streams are fused in a normalized residual space and mapped to the final forecast through a residual-to-last recombination.

B. Normalization and Series Decomposition

a) *Train-only data normalization*: Before feeding data into the network, we normalize the raw time series using statistics fitted on the *training split only* to avoid information leakage. In our implementation, all datasets use standardization; in all cases, scaler parameters are estimated on the training split and then applied to validation and test splits.

QuadFocusNet: Dual-Stream Forecasting with Adaptive Patching, Temporal Prompts, and Residual-to-Last Recomposition

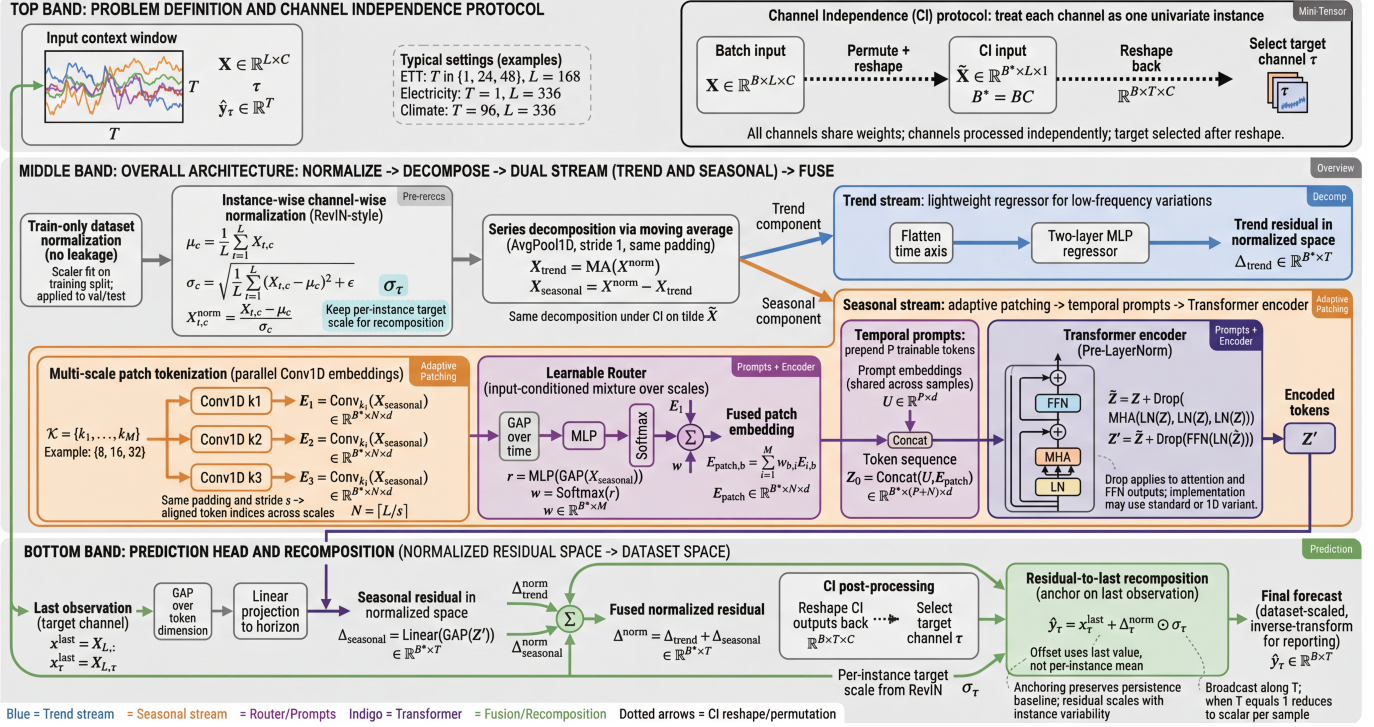


Fig. 2: **Overall Workflow of QuadFocusNet** We adopt channel-independent processing ($B^*=BC$), use RevIN-style normalization and moving-average decomposition, and forecast through a dual-stream design: a lightweight trend regressor for low-frequency variations and a seasonal Transformer driven by router-fused multi-scale patch tokens with temporal prompts. The two streams are fused in normalized residual space, reshaped back to $\mathbb{R}^{B \times T \times C}$, and recomposed with a residual-to-last formulation for the target channel: $\hat{y}_\tau = x_\tau^{\text{last}} + \Delta_\tau^{\text{norm}} \odot \sigma_\tau$. For more details about QuadFocusNet’s architecture, please refer to Figure 3.

b) *Instance-wise channel-wise normalization (RevIN-style):* We further apply an instance-wise, channel-wise normalization over the lookback window inside the model. For each sample and each channel c , we compute

$$\mu_c = \frac{1}{L} \sum_{t=1}^L X_{t,c}, \quad (2)$$

$$\sigma_c = \sqrt{\frac{1}{L} \sum_{t=1}^L (X_{t,c} - \mu_c)^2 + \epsilon},$$

where ϵ is a small constant for numerical stability. The normalized input is

$$X_{t,c}^{\text{norm}} = \frac{X_{t,c} - \mu_c}{\sigma_c} \quad (3)$$

We denote by σ_τ the resulting per-instance standard deviation for the target channel; it is later used in the residual-to-last recomposition.

c) *Series decomposition via moving average:* We apply a moving-average decomposition implemented by one-dimensional average pooling with a stride of 1 and the same padding. Let $\text{MA}(\cdot)$ denote this operator:

$$X_{\text{trend}} = \text{MA}(X^{\text{norm}}), \quad (4)$$

$$X_{\text{seasonal}} = X^{\text{norm}} - X_{\text{trend}}.$$

Under CI, the same decomposition is applied to $\tilde{X} \in \mathbb{R}^{B^* \times L \times 1}$.

C. Adaptive Patching with Learnable Router

a) *Multi-scale patch tokenization:* We generate tokens at multiple scales using parallel one-dimensional convolutional embeddings. Let $\mathcal{K} = \{k_1, \dots, k_M\}$ be a set of kernel sizes (e.g., {8, 16, 32}). For each k_i , we apply a Conv1D with stride s and same padding to the seasonal signal, producing

$$E_i = \text{Conv}_{k_i}(X_{\text{seasonal}}) \in \mathbb{R}^{B^* \times N \times d} \quad (5)$$

where d is the model dimension and N is the number of tokens. With same padding and stride s , $N = \lceil L/s \rceil$, so all E_i share aligned token indices.

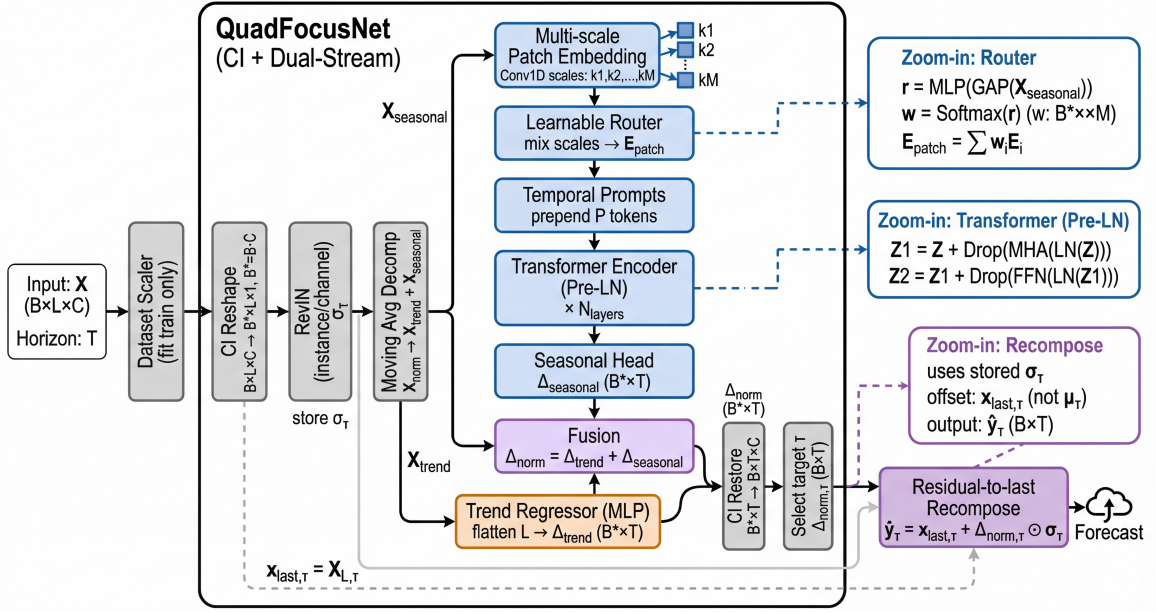
b) *Learnable router and patch fusion:* A lightweight router outputs mixture weights over scales conditioned on the input. Formally, the routing process is defined as follows:

$$r = \text{MLP}(\text{GAP}(X_{\text{seasonal}})), \quad (6)$$

$$w = \text{Softmax}(r).$$

Here $w \in \mathbb{R}^{B^* \times M}$ and $\sum_{i=1}^M w_{b,i} = 1$ for each instance b . The fused patch embedding is computed instance-wise:

$$E_{\text{patch},b} = \sum_{i=1}^M w_{b,i} E_{i,b}, \quad E_{\text{patch}} \in \mathbb{R}^{B^* \times N \times d} \quad (7)$$



(2).png

Fig. 3: **QuadFocusNet architecture.** We apply channel-independent processing ($B^*=BC$), RevIN-style normalization, and moving-average decomposition to obtain trend and seasonal components. The trend stream uses a lightweight MLP regressor, while the seasonal stream encodes **router-fused multi-scale** patch tokens with temporal prompts via a Transformer. Their fused normalized residual is reshaped back and recomposed with a residual-to-last anchor to produce $\hat{y}_T$.

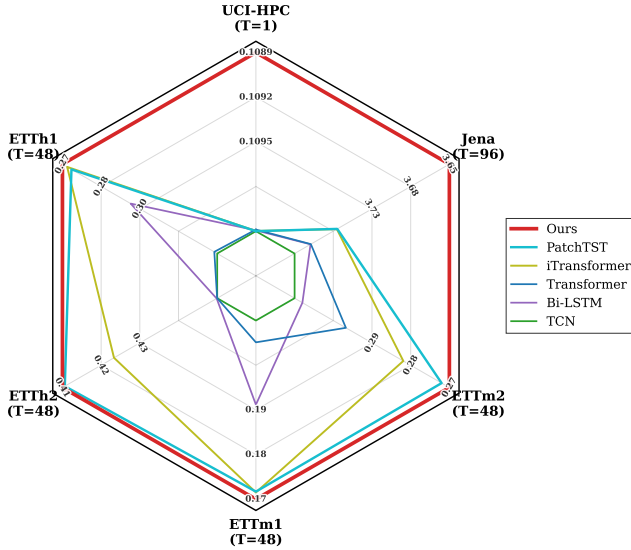


Fig. 4: Part of the performance comparison on multiple datasets. The radar chart shows that QuadFocusNet (Red) achieves competitive or superior results, demonstrating superior performance across diverse scenarios.

D. Temporal Prompts and Representation Learning

a) *Temporal prompts:* We prepend P trainable prompt tokens to the patch token sequence. Let $U \in \mathbb{R}^{P \times d}$ be the prompt embedding matrix shared across samples. Given

$E_{\text{patch}} \in \mathbb{R}^{B^* \times N \times d}$, the token sequence is:

$$Z_0 = \text{Concat}(U, E_{\text{patch}}) \in \mathbb{R}^{B^* \times (P+N) \times d} \quad (8)$$

b) *Transformer encoder with Pre-LayerNorm:* We adopt a Transformer block with pre-normalization for stable optimization:

$$\begin{aligned} \tilde{Z} &= Z + \text{Drop}\left(\text{MHA}(\text{LN}(Z), \text{LN}(Z), \text{LN}(Z))\right), \\ Z' &= \tilde{Z} + \text{Drop}\left(\text{FFN}(\text{LN}(\tilde{Z}))\right). \end{aligned} \quad (9)$$

$\text{Drop}(\cdot)$ denotes dropout applied to attention and feed-forward outputs (implementation may use either standard dropout or its 1D variant for stability on specific datasets).

E. Prediction Head and Recomposition

a) *Trend prediction:* After flattening the trend signal along the time axis, we apply a two-layer MLP to obtain a T -step residual prediction in the normalized space:

$$\Delta_{\text{trend}} \in \mathbb{R}^{B^* \times T}.$$

b) *Seasonal prediction from token representations:* From the Transformer output Z' , we obtain a global representation via global average pooling across the token dimension and map it to the horizon:

$$\Delta_{\text{seasonal}} = \text{Linear}\left(\text{GAP}(Z')\right) \in \mathbb{R}^{B^* \times T} \quad (10)$$

c) *Fusion in normalized residual space:* We fuse the two streams additively:

$$\Delta^{\text{norm}} = \Delta_{\text{trend}} + \Delta_{\text{seasonal}} \in \mathbb{R}^{B^* \times T}. \quad (11)$$

d) *Residual-to-last recombination*: Instead of predicting absolute values directly, we predict residuals and anchor them to the last observation. All recombination is performed in the dataset-scaled space, and we report final metrics after inverse-transforming the target channel. Let $\Delta_\tau^{\text{norm}} \in \mathbb{R}^{B \times T}$ denote the fused residual for the target channel after reshaping CI outputs. The final prediction is computed by anchoring on the last observation and re-scaling the residuals. Note that we use x_τ^{last} as the offset term instead of adding back the per-instance mean μ_τ . By applying broadcasting along the time dimension T , we obtain:

$$\hat{y}_\tau = x_\tau^{\text{last}} + \Delta_\tau^{\text{norm}} \odot \sigma_\tau, \quad (12)$$

where $\odot$ denotes element-wise multiplication with broadcasting, and $\hat{y}_\tau \in \mathbb{R}^{B \times T}$. For $T = 1$, the broadcasting reduces to scalar operations, and $\hat{y}_\tau$ is a scalar per sample. This residual-to-last formulation preserves the persistence baseline while allowing learned corrections that scale with instance-specific variability.

The overall workflow is shown in detail in Figure 2.

IV. EXPERIMENTS

We evaluate QuadFocusNet on three benchmark datasets covering distinct forecasting horizons. The experiments include: (1) *one-step-ahead prediction* ($T = 1$) on the UCI-HPC dataset; (2) *short- and mid-term forecasting* on ETT with $T \in \{1, 24, 48\}$; and (3) *long-term forecasting* on Jena Climate with $T = 96$.

A. Experimental Setup

1) *Datasets and Protocols*: We utilize three standard benchmarks with the following configurations:

- **UCI Household Power Consumption (UCI-HPC)**: A high-frequency dataset where we utilize a lookback window of $L = 336$ to predict the next step ($T = 1$).
- **Electricity Transformer Temperature (ETT)**: Including ETTh1, ETTh2, ETTm1, and ETTm2. We set the lookback window $L = 168$ and evaluate on horizons $T \in \{1, 24, 48\}$.
- **Jena Climate**: A meteorological dataset recorded at 10-minute intervals (aggregated hourly). We use $L = 336$ to forecast a long horizon of $T = 96$ steps.

Data Partitioning: All datasets are partitioned into 70% training, 10% validation, and 20% testing. To prevent information leakage, the data scaler is fitted solely on the training split.

Context-aware Evaluation: During validation and test phases, we strictly implement a context-aware mechanism that prepends input sequences with the last L time steps from the preceding split, ensuring valid historical windows for every target point.

2) *Implementation Details*: We employ the Adam optimizer with a learning rate of 10^{-4} and the Huber loss ($\delta = 1.0$) to enhance robustness against outliers. The early stopping patience is set to 8 epochs to prevent overfitting, with a maximum of 50 epochs to ensure model convergence.

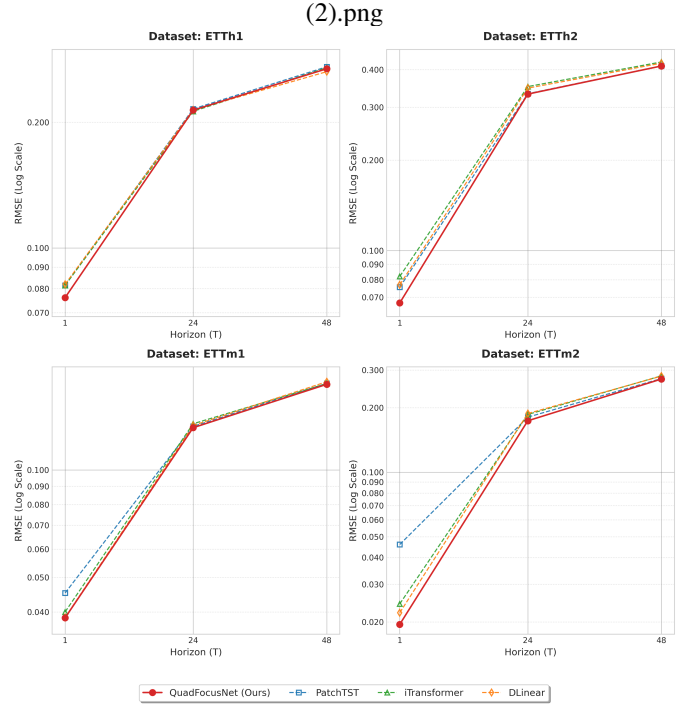


Fig. 5: Visualization of RMSE performance on ETTh1, ETTh2, ETTm1, and ETTm2 datasets. The vertical axis employs a logarithmic scale. QuadFocusNet consistently achieves the lowest error rates across prediction horizons of 1, 24, and 48 and demonstrates superior robustness compared to PatchTST, iTransformer, and DLinear.

- **Batch Size**: 128 for the UCI-HPC dataset; 32 for ETT and Climate datasets, considering different dataset sizes.
- **Regularization**: We apply a standard dropout of 0.2. Specifically, for the Climate dataset (long-term task), we replace standard dropout with `SpatialDropout1D` in the Transformer block to enforce feature-channel independence and prevent overfitting on long sequences.
- **Training Strategy**: Early stopping is employed with a patience of 8 epochs. The learning rate is reduced by a factor of 0.5 upon validation loss plateaus.

B. Results on UCI-HPC dataset

In this context, we evaluate our model’s ability to predict immediate future values by reporting the Root Mean Squared Error and Mean Absolute Error in the scaled space. Table I shows the performance of various models on the UCI-HPC dataset. QuadFocusNet achieves an RMSE of 0.1089 and outperforms all baseline models. Although the MAE of QuadFocusNet is slightly higher than that of Stacked GRU, our model still offers comparable performance.

C. Results on ETT datasets

In this setting, we evaluate the model’s robustness on the Electricity Transformer Temperature (ETT) datasets. These datasets include both hourly-recorded subsets (ETTh1, ETTh2)

and minute-recorded subsets (ETTh1, ETTh2), across time horizons of $T \in \{1, 24, 48\}$.

Table I summarizes the performance on the ETT datasets. QuadFocusNet’s superiority is particularly evident on ETTh2, where the model achieves the lowest error rates across all time horizons, notably reaching an RMSE of 0.4115 at $T = 48$. On ETTh1, our model achieves the best performance at horizons of 1 and 24, with an RMSE of 0.0761 at $T = 1$, and remains highly competitive against strong baselines like DLinear at the extended horizon of 48.

Regarding the minute-resolution datasets, QuadFocusNet demonstrates dominance across nearly all metrics, emphasizing its ability to handle high-frequency data fluctuations. On ETTm1 at a horizon of 48, our model achieves an RMSE of 0.1739, surpassing TimesNet (0.1741). This performance advantage is even more pronounced on ETTm2; notably, at $T = 1$, QuadFocusNet achieves an RMSE of 0.0195 compared to DLinear’s 0.0221. Additionally, the model maintains the lowest error rates across the longer horizons of 24 and 48, confirming its effectiveness in capturing detailed temporal dependencies.

We attribute this robustness to our model’s ability to dynamically shift attention between different patch scales: for high-frequency minute-level data, the router can prioritize finer-grained patches to capture rapid load fluctuations, whereas for smoother hourly data, it leverages larger receptive fields to model broader periodicities.

D. Results on Jena dataset

We further evaluate the model on the Jena Climate dataset with a significantly longer prediction horizon of 96 to test its stability in long-term forecasting. As shown in Table I, QuadFocusNet achieves an RMSE of 3.6504 and an MAE of 2.7993. This performance exceeds all baseline models, including PatchTST, which has an RMSE of 3.7878. The superior performance can be attributed to our multi-scale adaptive patching strategy.

V. ABLATION STUDY

To verify the contribution of each component in QuadFocusNet, we conduct systematic ablation studies by removing or replacing key modules. Specifically, we evaluate the impact of the decomposition and trend stream, multi-scale patching (versus single-scale variants), the learned router (versus uniform averaging), and auxiliary components such as prompt learning. Detailed numerical results are reported in Tables II, III, IV.

Across all evaluated settings, the full model achieves the lowest error. Notably, disabling the trend stream or removing the decomposition—trend pathway results in apparent performance degradation, underscoring the importance of the dual-stream design for managing non-stationarity. Additionally, the learned router consistently outperforms uniform averaging, highlighting the advantage of dynamic, context-aware scale weighting over static aggregation.

VI. CONCLUSION

In this paper, we propose QuadFocusNet, a unified framework that addresses the limitations of fixed-scale patching in multivariate time series forecasting. By combining series decomposition with a novel Multi-scale Adaptive Patching mechanism, our architecture effectively captures local variations while modeling global trends.

The core innovation is the Learnable Router. This component enhances feature extraction from a static process to a dynamic, context-aware routing system. The model then functions like a “variable-focus lens,” automatically selecting the most informative receptive fields for evolving temporal dynamics. Additionally, integrating Prompt Learning with strong normalization techniques (RevIN-style instance normalization) boosts robustness against non-stationarity and dataset-specific variability.

Extensive experiments on the UCI-HPC, ETT, and Climate datasets show that QuadFocusNet consistently performs well across different evaluation scenarios. These include effective one-step-ahead forecasting and long-horizon predictions. Our ablation studies also confirm that the model’s performance results from the combined effects of decomposition, adaptive fusion across levels, and prompt-enhanced representation. Overall, this work emphasizes the importance of adaptive granularity in modern forecasting and supports flexible, prompt-based time series architectures.

REFERENCES

- [1] A. Gasparin, S. Lukovic, and C. Alippi, “Deep learning for time series forecasting: the electric load case,” *CAAI Trans. Intell. Technol.*, vol. 7, no. 1, pp. 1–25, 2022.
- [2] C. Deb, F. Zhang, J. Yang, S. E. Lee, and K. W. Shah, “A review on time series forecasting techniques for building energy consumption,” *Renewable and Sustainable Energy Reviews*, vol. 74, pp. 902–924, 2017.
- [3] I. Koprinska, D. Wu, and Z. Wang, “Convolutional neural networks for energy time series forecasting,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, pp. 1–8, 2018.
- [4] X. Ren, X. Li, K. Ren, J. Song, Z. Xu, K. Deng, and X. Wang, “Deep learning-based weather prediction: a survey,” *Big Data Research*, vol. 23, p. 100178, 2021.
- [5] Y. Wang, Y. Xu, J. Yang, M. Wu, X. Li, L. Xie, and Z. Chen, “Fully-connected spatial-temporal graph for multivariate time-series data,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 14, pp. 15715–15724, 2024.
- [6] C. Ma, G. Dai, and J. Zhou, “Short-term traffic flow prediction for urban road sections based on time series analysis and LSTM_BILSTM method,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 5615–5624, 2021.
- [7] Z. Xu, J. Yuan, L. Yu, G. Wang, and M. Zhu, “Machine learning-based traffic flow prediction and intelligent traffic management,” *International Journal of Computer Science and Information Technology*, vol. 2, no. 1, pp. 18–27, 2024.
- [8] J. Contreras, R. Espinola, F. J. Nogales, and A. J. Conejo, “ARIMA models to predict next-day electricity prices,” *IEEE Trans. Power Syst.*, vol. 18, no. 3, pp. 1014–1020, 2003.
- [9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [10] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, “Transformers in time series: A survey,” *arXiv preprint arXiv:2202.07125*, 2022.
- [11] Y. Nie, “A time series is worth 64 words: long-term forecasting with transformers,” *arXiv preprint arXiv:2211.14730*, 2022.
- [12] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, “TimesNet: temporal 2D-variation modeling for general time series analysis,” *arXiv preprint arXiv:2210.02186*, 2022.

TABLE I: Comprehensive forecasting results across all datasets. We evaluate the models on UCI-HPC ($T = 1$), Jena Climate ($T = 96$), and four ETT datasets ($T \in \{1, 24, 48\}$). **Results for Jena Climate are reported using unscaled metrics, while the others are presented in scaled space.** The best results are highlighted in **red**, and the second-best are **blue**.

Dataset	T	Metric	Models										
			TCN	LSTM	Stacked-GRU	Bi-LSTM	LSTM-Attention	Transformer	DLinear	PatchTST	iTransformer	TimesNet	Ours
UCI-HPC	1	RMSE	0.1152	0.1148	0.1112	0.1129	0.1141	0.1135	0.1138	0.1144	0.1137	0.1202	0.1089
		MAE	0.0825	0.0812	0.0761	0.0785	0.0810	0.0792	0.0803	0.0823	0.0795	0.0864	0.0776
ETTh1	1	RMSE	0.0895	0.0858	0.0937	0.0824	0.0882	0.0865	0.0819	0.0814	0.0814	0.0902	0.0761
		MAE	0.0642	0.0607	0.0668	0.0595	0.0635	0.0618	0.0576	0.0578	0.0576	0.0656	0.0519
	24	RMSE	0.2315	0.2252	0.2189	0.2236	0.2304	0.2285	0.2142	0.2154	0.2125	0.2147	0.2139
		MAE	0.1785	0.1720	0.1678	0.1682	0.1762	0.1745	0.1585	0.1596	0.1583	0.1611	0.1582
	48	RMSE	0.3350	0.3222	0.2976	0.2963	0.3285	0.3156	0.2645	0.2717	0.2703	0.2809	0.2688
		MAE	0.2612	0.2501	0.2249	0.2287	0.2588	0.2455	0.2005	0.2050	0.2048	0.2184	0.2037
ETTh2	1	RMSE	0.0855	0.0829	0.0831	0.0796	0.0842	0.0815	0.0770	0.0756	0.0821	0.1053	0.0670
		MAE	0.0638	0.0612	0.0613	0.0604	0.0625	0.0598	0.0550	0.0551	0.0608	0.0777	0.0480
	24	RMSE	0.4420	0.4318	0.3759	0.4262	0.4385	0.4250	0.3473	0.3328	0.3515	0.3669	0.3318
		MAE	0.3550	0.3431	0.2781	0.3299	0.3485	0.3310	0.2543	0.2401	0.2597	0.2733	0.2385
	48	RMSE	0.5620	0.5417	0.5431	0.5579	0.5580	0.5350	0.4205	0.4120	0.4244	0.4435	0.4115
		MAE	0.4450	0.4296	0.4128	0.4604	0.4385	0.4210	0.3184	0.3110	0.3210	0.3350	0.3106
ETTm1	1	RMSE	0.0525	0.0494	0.0828	0.0518	0.0530	0.0512	0.0386	0.0452	0.0400	0.0418	0.0385
		MAE	0.0385	0.0345	0.0687	0.0373	0.0392	0.0365	0.0258	0.0317	0.0271	0.0289	0.0258
	24	RMSE	0.1425	0.1388	0.1325	0.1320	0.1450	0.1395	0.1333	0.1325	0.1348	0.1330	0.1314
		MAE	0.1020	0.0975	0.0925	0.0918	0.1055	0.0985	0.0897	0.0895	0.0944	0.0913	0.0890
	48	RMSE	0.2050	0.1973	0.1797	0.1907	0.2085	0.1950	0.1768	0.1749	0.1748	0.1741	0.1739
		MAE	0.1520	0.1442	0.1313	0.1362	0.1550	0.1425	0.1239	0.1222	0.1221	0.1195	0.1215
ETTm2	1	RMSE	0.0435	0.0412	0.0609	0.0486	0.0450	0.0373	0.0221	0.0460	0.0243	0.0251	0.0195
		MAE	0.0335	0.0312	0.0453	0.0377	0.0345	0.0268	0.0155	0.0353	0.0173	0.0182	0.0142
	24	RMSE	0.2410	0.2354	0.1943	0.2074	0.2455	0.1960	0.1880	0.1808	0.1861	0.1980	0.1739
		MAE	0.1650	0.1603	0.1270	0.1355	0.1710	0.1332	0.1185	0.1153	0.1201	0.1276	0.1101
	48	RMSE	0.3750	0.3642	0.3133	0.3446	0.3820	0.3012	0.2811	0.2738	0.2819	0.3236	0.2724
		MAE	0.2750	0.2649	0.2210	0.2463	0.2810	0.2144	0.1835	0.1815	0.1876	0.2135	0.1805
Jena	96	RMSE	4.6589	4.3196	4.1803	3.8833	5.1681	3.8814	3.8207	3.7878	3.7888	3.8452	3.6504
		MAE	3.6956	3.3107	3.2018	2.9777	4.0627	2.9805	2.9393	2.9323	2.9379	3.0221	2.7993

TABLE II: Ablation study on UCI-HPC (Short-term, $T = 1$) using scaled metrics. The table compares Patching Strategies (Left) and Architectural Components (Right). The Full Model (highlighted in gray) achieves the lowest error

Patching & Router Modules			Architecture & Baselines		
Model Variant	RMSE	MAE	Model Variant	RMSE	MAE
<i>Patching Strategy Analysis</i>			<i>Component Analysis</i>		
Fixed Patching ($P = 16$)	0.1203	0.0838	w/o Trend Stream	0.1236	0.0845
Adaptive Patching [8]	0.1235	0.0840	w/o Decomposition	0.1216	0.0854
Adaptive Patching [16]	0.1205	0.0833	w/o Prompt Learning	0.1183	0.0818
Adaptive Patching [16, 32]	0.1175	0.0817	w/o Channel Indep.	0.1176	0.0818
<i>Router Mechanism</i>			<i>Baselines</i>		
w/ Uniform Router	0.1185	0.0822	Persistence (Naive)	0.1370	0.0877
Full Model (Ours)	0.1089	0.0776	Full Model (Ours)	0.1089	0.0776

- [13] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 9, pp. 11121–11128, 2023.
- [14] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.
- [15] C. A. Sims, "Macroeconomics and reality," *Econometrica*, pp. 1–48, 1980.
- [16] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with LSTM," *Neural Comput.*, vol. 12, no. 10, pp. 2451–2471, 2000.
- [17] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP)*, 2014.
- [18] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, 2019.
- [19] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.
- [20] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 156–165, 2017.
- [21] C. Lea, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks: A unified approach to action segmentation," in *Proc. European*

TABLE III: Detailed ablation study across four ETT datasets with step-wise breakdown ($H = 1, 24, 48$). The Full Model consistently achieves the lowest error. Significant performance drops in ablation variants demonstrate the necessity of each component

Model Variant	ETTh1 (RMSE)			ETTh2 (RMSE)			ETTm1 (RMSE)			ETTm2 (RMSE)		
	1	24	48	1	24	48	1	24	48	1	24	48
Patching Strategy												
Fixed Patch ($P = 16$)	0.0766	0.2161	0.2737	0.0685	0.3427	0.4185	0.0385	0.1316	0.1752	0.0195	0.1750	0.2745
Adaptive [8]	0.0766	0.2152	0.2739	0.0764	0.3492	0.4387	0.0385	0.1316	0.1742	0.0218	0.1876	0.2802
Adaptive [16]	0.0777	0.2156	0.2718	0.0674	0.3422	0.4117	0.0386	0.1314	0.1747	0.0199	0.1741	0.2744
Adaptive [32]	0.0763	0.2154	0.2736	0.0678	0.3490	0.4172	0.0386	0.1319	0.1748	0.0201	0.1772	0.2731
Router Mechanism												
w/ Uniform Router	0.0774	0.2147	0.2698	0.0731	0.3381	0.4119	0.0386	0.1325	0.1748	0.0207	0.1754	0.2761
Component Analysis												
w/o Prompt Learning	0.0773	0.2152	0.2698	0.0686	0.3363	0.4156	0.0385	0.1317	0.1745	0.0200	0.1750	0.2740
w/o Trend Stream	0.0770	0.2176	0.2828	0.0699	0.3454	0.4295	0.0386	0.1419	0.1841	0.0197	0.2080	0.3197
w/o Decomposition	0.0774	0.2152	0.2750	0.0706	0.3711	0.4350	0.0386	0.1343	0.1785	0.0220	0.1940	0.2889
w/o CI (Target-only)	0.0767	0.2155	0.2718	0.0682	0.3394	0.4225	0.0385	0.1316	0.1737	0.0200	0.1762	0.2780
Full Model (Ours)	0.0761	0.2139	0.2688	0.0670	0.3318	0.4115	0.0385	0.1314	0.1739	0.0195	0.1739	0.2724

TABLE IV: Comprehensive ablation study on Jena Climate Dataset (here $T = 96$). Results demonstrate the effectiveness of each component using **unscaled** metrics. The Full Model (bottom) outperforms all ablated variants

Patching & Router Modules			Architecture & Regularization		
Model Variant	RMSE	MAE	Model Variant	RMSE	MAE
Patching Strategy Analysis			Component Analysis		
Fixed Patching ($P = 16$)	3.8945	3.0120	w/o Trend Stream	4.1500	3.2405
Adaptive Patching [8]	4.0521	3.1450	w/o Decomposition	4.0820	3.1850
Adaptive Patching [16]	3.9840	3.0920	w/o Residual-to-Last	3.9108	3.0450
Adaptive Patching [16, 32]	3.8510	2.9880	w/o Prompt Learning	3.8450	2.9610
Router Mechanism			Regularization		
w/ Uniform Router	3.8655	2.9940	w/o SpatialDropout1D	3.9512	3.0844
Full Model (Ours)	3.6504	2.7993	Full Model (Ours)	3.6504	2.7993

Conf. Comput. Vis. (ECCV), pp. 47–54, 2016.

- [22] Y. A. Farha and J. Gall, “MS-TCN: Multi-stage temporal convolutional network for action segmentation,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 3575–3584, 2019.
- [23] Y. He and J. Zhao, “Temporal convolutional networks for anomaly detection in time series,” in *J. Phys.: Conf. Ser.*, vol. 1213, no. 4, p. 042050, 2019.
- [24] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, “A transformer-based framework for multivariate time series representation learning,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. (KDD)*, pp. 2114–2124, 2021.
- [25] Y. Chen, K. Ren, Y. Wang, Y. Fang, W. Sun, and D. Li, “Contiformer: Continuous-time transformer for irregular time series modeling,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, pp. 47143–47175, 2023.
- [26] N. Wu, B. Green, X. Ben, and S. O’Banion, “Deep transformer models for time series forecasting: The influenza prevalence case,” *arXiv preprint arXiv:2001.08317*, 2020.
- [27] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: beyond efficient transformer for long sequence time-series forecasting,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 12, pp. 11106–11115, 2021.
- [28] H. Wu, J. Xu, J. Wang, and M. Long, “Autoformer: decomposition transformers with auto-correlation for long-term series forecasting,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, pp. 22419–22430, 2021.
- [29] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, “FEDformer: frequency enhanced decomposed transformer for long-term series forecasting,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 27268–27286, 2022.
- [30] S. Liu, H. Yu, C. Liao, J. Li, W. Lin, A. X. Liu, and S. Dustdar, “Pyraformer: low-complexity pyramidal attention for long-range time series modeling and forecasting,” 2022.
- [31] Y. Zhang and J. Yan, “Crossformer: transformer utilizing cross-dimension dependency for multivariate time series forecasting,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2023.
- [32] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, “iTransformer: inverted transformers are effective for time series forecasting,” *arXiv preprint arXiv:2310.06625*, 2023.
- [33] V. Ekambaram, A. Jati, N. Nguyen, P. Sinthong, and J. Kalagnanam, “TSMixer: lightweight MLP-mixer model for multivariate time series forecasting,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. (KDD)*, pp. 459–469, 2023.
- [34] F. M. Shiri, T. Perumal, N. Mustapha, and R. Mohamed, “A comprehensive overview and comparative analysis on deep learning models: CNN, RNN, LSTM, GRU,” *arXiv preprint arXiv:2305.17473*, 2023.
- [35] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, “Stock price prediction using the ARIMA model,” in *Proc. UKSim-AMSS Int. Conf. Comput. Model. Simul.*, pp. 106–112, 2014.