RAFU-FORMER: RAW-INPUT ADAPTIVE FUSED TRANSFORMER FOR TIME SERIES FORCASTING

Linghui SHEN
Department of Coumputing
linghui.shen@connet.polyu.hk

Lingru HUANG
Department of Coumputing
lingru.huang@connect.polyu.hk

Rou Xin TAN
Department of Coumputing
estella.tan@connect.polyu.hk

ABSTRACT

Multivariate Time Series Forecasting (MTSF) is crucial for many applications but faces challenges in efficiently modeling complex cross-variable dependencies while maintaining interpretability. Standard Transformer models often suffer from quadratic complexity in time steps and variables, while traditional decomposition methods lack adaptability to dynamic interdependencies. To address these limitations, we propose Rafu-Former (Raw-input Adaptive Fused Transformer), an efficient and interpretable architecture for MTSF. Rafu-Former decouples cross-variable modeling from temporal pattern extraction using three key components: First, a Cross-Variable Mixer that operates on compressed variate tokens via multi-head attention to efficiently capture inter-variable relationships. Next, a Dynamic Series Adapter that decomposes each series into interpretable trend, seasonal, and residual bases, dynamically weighting them based on attention-derived cross-variable context. Finally, a Variable Predictor with a partially shared structure for parameter efficiency and capturing both shared and variable-specific patterns. This design significantly reduces computational complexity while enhancing model adaptability and interpretability. Experiments on benchmark datasets demonstrate that Rafu-Former achieves state-of-the-art forecasting accuracy, particularly for medium-length horizons, outperforming existing methods like BasisFormer by effectively balancing efficiency, interpretability, and adaptive cross-variable dependency modeling. The results suggest that traditional decomposition methods, when properly enhanced, remain competitive foundations for modern forecasting systems.

Keywords Multivariate Time Series Forecasting · Cross-Variable Modeling · Dynamic Basis Weighting · Interpretable decomposition · Parameter-efficient attention

1 Introduction

Multivariate Time Series Forecasting (MTSF) plays a crucial role in numerous real-world applications, ranging from energy consumption prediction and traffic flow management to financial market analysis and weather forecasting. Effectively forecasting future values across multiple correlated time series requires models capable of capturing both complex temporal dynamics within individual series and intricate dependencies *between* different series.

However, modern MTSF faces significant challenges. Firstly, traditional statistical methods often struggle with the high dimensionality and non-linear relationships inherent in contemporary datasets, which can involve hundreds of variables (see Section 2.1). Secondly, while deep learning models, particularly Transformer-based architectures, have shown promise, standard implementations often suffer from prohibitive computational complexity, scaling poorly with both the number of time steps (T) and variables (V) due to their dual attention mechanisms (see Sections 2.2, 3.1). Thirdly, effectively modelling the relationships *across* variables (cross-variable dependencies) remains difficult, especially when these relationships are dynamic. Approaches like graph neural networks (GNNs) have been explored (e.g., MTGNN

[Wu et al., 2020], StemGNN [Cao et al., 2021]), but can struggle with adaptability (see Section 2.2). Finally, maintaining model interpretability—understanding *how* predictions are made by decomposing series into components like trend and seasonality—is vital for actionable insights, yet challenging to balance with predictive accuracy and the modelling of cross-variable interactions (see Section 2.3).

Previous research has attempted to address these issues. Transformer variants like Informer [Zhou et al., 2021] and Autoformer [Wu et al., 2022] introduced sparse attention or decomposition mechanisms to reduce complexity, while PatchTST [Nie et al., 2023] employed patching for local temporal patterns (see Section 2.1). For interpretability, classical methods like STL [Skipper and Josef, 2010] offer clear decomposition but lack adaptability and struggle with multivariate interdependencies. Neural decomposition methods like BasisFormer [Ni et al., 2023] improved on this by using learnable bases but relied on fixed projections that may not capture variable-specific context adequately (see Section 2.3). More recently, iTransformer [Liu et al., 2024] highlighted the effectiveness of variate-centric representation learning, treating entire variable series as tokens to improve efficiency (see Sections 3, 3.1).

To overcome these limitations, we introduce **Rafu-Former** (Raw-input Adaptive Fused Transformer), a novel architecture designed for efficient, interpretable, and accurate MTSF. Inspired by the success of variate-centric approaches [Liu et al., 2024], Rafu-Former tackles the core challenges by decoupling the modelling of cross-variable relationships from temporal pattern extraction through three synergistic components: First, the Cross-Variable Mixer efficiently models relationships between variables by operating on compressed variate tokens generated from the raw input, significantly reducing computational complexity compared to standard dual-attention Transformers (see Section 3.1). Next, the Dynamic Series Adapter goes beyond static decomposition methods like STL or fixed neural bases like BasisFormer [Ni et al., 2023], decomposing each time series into interpretable trend, seasonal, and residual components. Crucially, it uses attention-derived cross-variable relationships (from the Mixer) to dynamically weight these bases for each specific variable (see Section 3.2). Finally, the Variable Predictor employs a partially shared architecture, balancing the learning of common temporal patterns across all variables with variable-specific prediction heads, achieving parameter efficiency while capturing unique variate characteristics (see Section 3.3).

Rafu-Former offers several advantages. By processing variate tokens, it drastically reduces the computational burden ($O(T^2V^2)$ for standard Transformers is avoided) associated with modelling high-dimensional time series (see Section 3.1). Its dynamic basis weighting mechanism enhances interpretability while allowing the model to adapt decomposition to specific variable contexts and their interdependencies, improving upon static methods like STL and fixed-basis approaches like BasisFormer (see Sections 2.3, 3.2). The partially shared predictor provides a favorable trade-off between parameter efficiency and the ability to model individual variable dynamics, outperforming strictly shared or fully independent models (see Sections 3.3, 4.2). Experimental results demonstrate that Rafu-Former achieves outstanding performance, particularly in medium-horizon forecasting, outperforming models like BasisFormer significantly (see Section 4.2). In summary, the key contributions of our work comprise:

- We propose a variate-centric attention mechanism that transforms raw time series into compressed tokens, enabling efficient modeling of cross-variable dependencies without the quadratic complexity of standard Transformers. This design preserves the interpretability of attention weights while scaling to hundreds of variables.
- We develop a dynamic decomposition framework that automatically adapts basis weights (trend/seasonal/residual) according to learned cross-variable relationships. The attention-guided weighting provides both interpretable components and improved accuracy over static decomposition methods.
- We validate the effectiveness of our proposed method through extensive comparisons against three foundational neural network models and a state-of-the-art time series forecasting approach, supported by detailed ablation analysis. The results demonstrate consistent performance advantages in electricity demand forecasting, especially for medium-range horizons (96–192 time steps). To further investigate the model's capabilities, we also incorporate pre-training experiments as part of the evaluation.

This report is structured as follows: Section 2 discusses related work in MTSF, cross-variable modelling, and interpretable decomposition. Section 3 details the architecture and components of Rafu-Former. Section 4 presents the experimental setup, results, and ablation studies comparing Rafu-Former against baseline models. Finally, Section 5 concludes the report and discusses potential future work

2 Related Works

The Rafu-Former model was aimed at advancing multivariate time series forecasting by increasing its efficiency in modelling cross-variable dependencies and maintaining interpretability through adaptive temporal pattern extraction. The combination of Cross-Variable Mixer, Dynamic Series Adapter and Variable Predictor modules were the main components of its architecture, which includes foundational techniques like attention, traditional decomposition

method and recent deep learning advancements. This section reviews related work across these domains, explaining Rafu-Former's position in the landscape of time series forecasting.

Multivariate Time Series Forecasting Models The majority of traditional statistical methods excel in low-dimensional settings but struggle with high-dimensional data and nonlinear dependencies. Their applicability to modern datasets that contain more than a hundred correlated variables is limited. Deep learning played an important role in multivariate forecasting. It emphasises temporal and cross-variable dependencies through various architectures. For example, the Transformer-based Models. Sparse attention was introduced in Informer Zhou et al. [2021] to reduce complexity, with Autoformer Wu et al. [2022] incorporating decomposition into attention. According to PatchTST Nie et al. [2023], patching was used to focus on local temporal patterns, receiving strong results on specific datasets. However, this dual attention mechanism suffers from high complexity, causing a large number of time steps and a large number of variables. Rafu-Former is able to solve this by reducing complexity through the Cross-Variable Mixer, which operates on compressed variate tokens. It is proven that higher efficiency can be achieved through the modelling of cross-variable relationships.

Cross-Variable Dependency Modelling Modelling relationships across variables is crucial in multivariate forecasting, especially for physical interaction variables. Some models integrate attention mechanisms such as Standard Transformers Vaswani et al. [2023]to model both temporal and cross-variable dependencies, but they suffer from the number of time steps and variables. Nevertheless, Rafu-Former made significant improvement over dual-attention transformers by Cross-Variable Mixer that uses multi-head attention on compressed variate embeddings, efficiency quantifying pairwise relationships. Other than attention mechanism, MTGNN Wu et al. [2020]and StemGNN Cao et al. [2021]use graph neural networks to model variable relationships, with the former learning dynamic graph and the latter combining spectral convolution with temporal attention. However, the results show that the graph structures produced have little to no adaptability to evolving relationships. With this being said, graph-based methods do face some issues in multivariate forecasting. The adaptive attention in Rafu-Former helps to avoid explicit graph assumptions, providing a better and flexible approach to dynamic dependency modelling.

Interpretable Time Series Decomposition In applications that require actionable insight, interpretability plays an important role in understanding the temporal pattern. Some classical methods were used in the earlier days, such as the Seasonal-Trend decomposition using Loess (STL) Skipper and Josef [2010]. It decomposes the series into trend, seasonal and residual components for simplicity and interpretability. The results remain static and fail for cross-variable interdependencies, limiting the effectiveness in multivariate settings. Neural decomposition was used in later days. BasisFormer Ni et al. [2023] employs interpretable bases via self-supervised learning and managed to improve 11-15% of specific datasets. Nonetheless, BasisFormer's fixed basis projections could not project variable-specific context. Rafu-Former advances this by adding attention-derived cross-variable relationships to weight trend, seasonal and residual bases. This is to emphasise on seasonal patterns of variates, bridging classical interpretability with neural adaptability to improve on static and fixed neural methods.

3 Raw-Input Adaptive Fused Former

Modern multivariate time series forecasting faces two fundamental challenges: (1) effectively modeling cross-variable dependencies without sacrificing computational efficiency, and (2) maintaining interpretability while adapting to variable-specific temporal patterns. Inspired by the success of iTransformer Liu et al. [2024] in variate-centric representation learning, we propose RafuFormer that addresses these challenges through three synergistic components, as illustrated in Figure 1.

The key insight is to decouple the modeling of cross-variable relationships from temporal pattern extraction. Given normalized input $\mathbf{X}_{\text{norm}} \in \mathbb{R}^{B\times T\times V}$, the model first generates compressed variate tokens through the Cross-Variable Mixer, then decomposes each variate's series into interpretable bases with dynamically learned weights in the Dynamic Series Adapter, and finally produces predictions via a partially shared architecture in the Variable Predictor. This design philosophy shares similarities with BasisFormer Ni et al. [2023] in maintaining interpretable components while introducing crucial innovations in dynamic basis weighting.

Given a multivariate time series $\mathbf{X} \in \mathbb{R}^{B\times T\times V}$ with B batches, T time steps, and V variables, our goal is to predict future values $\mathbf{Y} \in \mathbb{R}^{B\times T'\times V}$. The RafuFormer architecture processes the input through three key stages:

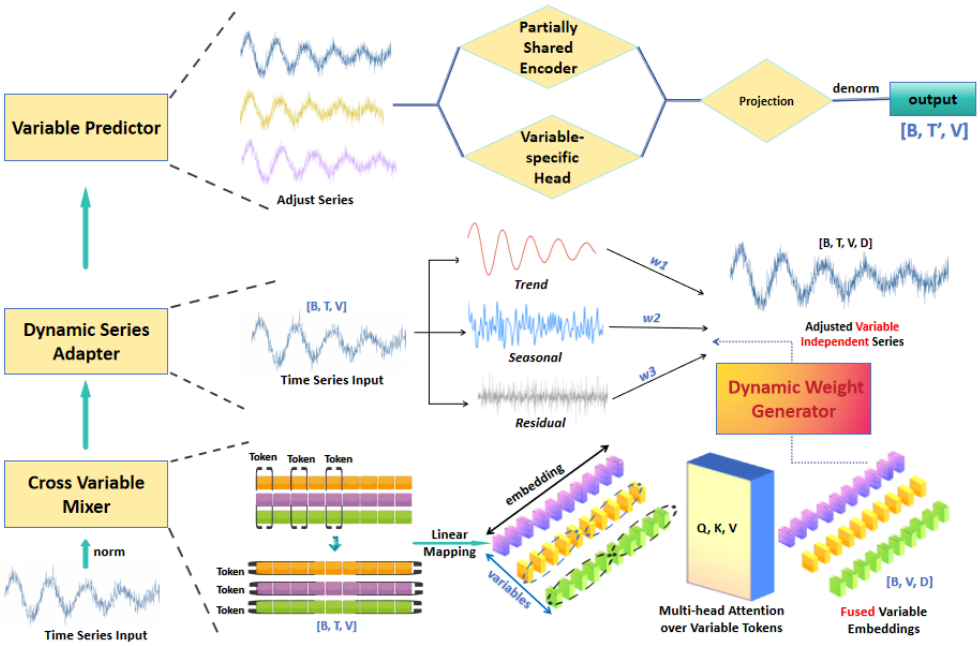

Figure 1: RafuFormer architecture. (a) The Cross-Variable Mixer (lower) computes attention weights between variable tokens, which generates variable sequence after fusion and enhancement. (b) The Dynamic Series Adapter (middle) decomposes inputs into interpretable bases. merging the dynamic weights to obtain new time series. (c) The Variable Predictor (upper) uses shared encoder techniques to variable independently generate final forecasts.

$$\begin{aligned} \mathbf{E}', \mathbf{A} &= \text{CrossVarMixer}(\mathbf{X}_{\text{norm}}) \\ \tilde{\mathbf{X}}, \mathbf{W} &= \text{DynamicAdapter}(\mathbf{X}_{\text{norm}}, \mathbf{E}') \\ \hat{\mathbf{Y}} &= \text{VariablePredictor}(\tilde{\mathbf{X}}) \end{aligned} \tag{1}$$

where $\mathbf{X}_{\text{norm}}$ is normalized input, $\mathbf{E}'$ denotes variable embeddings after attention fusion, $\mathbf{A}$ the attention matrix, $\tilde{\mathbf{X}}$ the adjusted series, and $\mathbf{W}$ the basis weights.

3.1 Cross-Variable Mixer Module

The computational bottleneck of standard Transformers in multivariate forecasting stems from their dual attention mechanism—simultaneously modeling $O(T^2)$ temporal dependencies and $O(V^2)$ cross-variable relationships, resulting in prohibitive $O(T^2V^2)$ complexity. This becomes particularly acute when handling long historical windows (large T) with hundreds of correlated variates (large V), as commonly encountered in power grid operations **?**. Recent work by Liu et al. [2024] demonstrated that treating entire variate series as atomic tokens can circumvent this issue while preserving predictive accuracy.

Building on this insight, our mixer module adopts a focused strategy: instead of processing raw time points, it first condenses each variate's temporal history into a compact embedding through adaptive averaging. Specifically, the initial mean pooling operation

$$\mathbf{E} = \text{MLP}\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{X}_{\text{norm}}^{(t)}\right) \in \mathbb{R}^{B\times V\times d} \tag{2}$$

These embeddings then interact through multi-head attention to capture pairwise variate relationships:

$$\mathbf{E}', \mathbf{A} = \text{MHA}(\mathbf{E}, \mathbf{E}, \mathbf{E}) \tag{3}$$

The attention Vaswani et al. [2023] matrix $\mathbf{A} \in \mathbb{R}^{B\times h\times V\times V}$ (where h is number of heads) explicitly quantifies variable relationships. Then operates exclusively on these compressed representations, enabling efficient modeling of variate relationships without sacrificing temporal awareness. This approach proves particularly effective in domains like electricity load forecasting, where variables (e.g., temperature, humidity, calendar features) exhibit stable physical relationships that persist across time scales.

3.2 Dynamic Series Adapter Module

The conventional STL decomposition framework Cleveland and Cleveland [1990] operates under the fundamental assumption that each time series can be decomposed into trend, seasonal, and residual components through fixed basis functions. While this approach provides interpretability, it fails to account for the complex interdependencies present in modern multivariate systems. Our dynamic adapter module introduces a paradigm shift by incorporating cross-variable contextual information into the basis decomposition process.The key innovation is the use of attention-derived variable relationships from the Mixer module to adaptively recalibrate the importance of each basis component for individual variates. This mechanism enables what we term "cross-variable basis modulation" - where the decomposition of a target variate's time series is consciously influenced by the characteristics of related variables. This represents a advancement over static decomposition methods, as the basis weights $\mathbf{W} \in \mathbb{R}^{B\times V\times 3}$ become functions of both the target variate's own history and its learned relationships with other variables.

The mathematical formulation of this process ensures that the adjusted time series $\tilde{\mathbf{X}}$ not only reflects the intrinsic patterns of each variate but also incorporates cross-variable contextual information. This is achieved through the mechanism that dynamically balances between the original STL-style decomposition and the attention-modulated version. The resulting representation maintains the interpretability of traditional decomposition methods while gaining the adaptability of modern neural approaches, effectively bridging the gap between classical time series analysis and contemporary multivariate forecasting techniques.

The module first generates three interpretable bases for each variate through complementary approaches:

$$\mathbf{B}^{\text{trend}} = \text{Conv1D}(\mathbf{X}_{\text{norm}}, k = 7) \tag{4}$$

$$\mathbf{B}^{\text{seasonal}} = \mathcal{F}^{-1}\left(|\mathcal{F}(\mathbf{X}_{\text{norm}})| \odot e^{j\angle\mathcal{F}(\mathbf{X}_{\text{norm}})}\right) \tag{5}$$

$$\mathbf{B}^{\text{residual}} = \text{Conv1D}(\mathbf{X}_{\text{norm}} - \mathbf{B}^{\text{trend}} - \mathbf{B}^{\text{seasonal}}) \tag{6}$$

Unlike static STL decomposition and equally weighted combination, the fusion weights are dynamically adapted using the attention-derived variable relationships from the Mixer module to adaptively recalibrate the relative importance of each basis component for every individual variate.

$$\mathbf{W} = \text{Softmax}(\text{MLP}(\mathbf{E}')) \in \mathbb{R}^{B\times V\times 3} \tag{7}$$

This design allows the model to, for instance, emphasize seasonal patterns for temperature-sensitive variates while focusing on trend components for economic indicators. The final adjusted series combines these weighted bases after dimension enhancement:

$$\tilde{\mathbf{X}} = \sum_{k=1}^{3} \mathbf{W}_{:,:,k} \odot \phi(\mathbf{B}^{(k)}) \tag{8}$$

where ϕ is a linear projection. The adapter's output preserves the interpretability of traditional decomposition methods while achieving the adaptability of modern neural approaches—an advancement over BasisFormer's fixed basis projection.

3.3 Variable Predictor Module

The design of prediction modules for multivariate time series forecasting must carefully balance two competing objectives: leveraging shared temporal patterns that exist across multiple variables while preserving the unique characteristics of individual variates. Recent work in Nie et al. [2023] has demonstrated the effectiveness of parameter sharing in reducing model complexity, while Wang et al. [2021] has shown the benefits of maintaining variable-specific

representations. Our predictor module synthesizes these perspectives through a novel architecture that automatically learns the optimal balance between shared and independent components.

At the core of this design is the recognition that different temporal features exhibit varying degrees of transferability across variables. The shared encoder component captures universal patterns that are consistently relevant across all variates, such as fundamental periodicities and system-wide trends. These shared features are then adaptively combined with variable-specific transformations through a learned projection mechanism. This approach is theoretically grounded in the concept of modular neural networks Happel and Murre [1994], where the model architecture explicitly encodes assumptions about which aspects of the data generation process are common across variables and which are unique.

The shared encoder first extracts common temporal features:

$$\mathbf{H} = \text{ReLU}(\mathbf{W}_s\text{Flatten}(\tilde{\mathbf{X}})) \in \mathbb{R}^{B\times V\times 256} \tag{9}$$

This shared representation captures universal patterns (e.g., diurnal cycles in electricity demand) that benefit from cross-variate learning. Variable-specific heads then tailor these features to individual variates:

$$\hat{\mathbf{Y}}_v = \mathbf{W}_v^{\text{pred}}\mathbf{H}_v + \mathbf{b}_v, \quad \forall v \in \{1, ..., V\} \tag{10}$$

This architecture achieves superior parameter efficiency compared to fully independent models while outperforming strictly shared approaches—particularly beneficial when dealing with hundreds of correlated variates as in power grid forecasting. The final output is rescaled to original value ranges:

$$\hat{\mathbf{Y}} = \hat{\mathbf{Y}} \odot \sigma + \mu \tag{11}$$

The mathematical implementation of this concept ensures efficient parameter utilization while maintaining the flexibility to capture variable-specific dynamics. The shared encoder operates on the basis-adjusted time series $\tilde{\mathbf{X}}$, extracting features that are subsequently processed by variable-specific prediction heads. This hierarchical structure naturally emerges from the probabilistic interpretation of multivariate time series forecasting, where the prediction for each variate can be viewed as conditionally dependent on both shared latent states and variable-specific parameters. The resulting architecture achieves superior generalization performance compared to fully shared or completely independent alternatives, as demonstrated in our experimental evaluation.

Loss Functions The training objective combines:

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda_1\mathcal{L}_{\text{smooth}} \tag{12}$$

where $\mathcal{L}_{\text{pred}}$ measures the model's prediction accuracy, while $\mathcal{L}_{\text{smooth}}$ encourages smoother outputs. The parameter λ_1 (set to 1 in experiments) balances the importance of these two components, guiding the model to make accurate yet consistent predictions.

4 Experiments

To evaluate the performance of the Rafu-Former model for both univariate and multivariate time series forecasting, we conducted comprehensive experiments using the Electricity dataset. This section details experimental setup, evaluation metric, results and ablation studies to highlight Rafu-Former's advancements in efficiency, interpretability and predictive accuracy.

Experimental Setup The experiments are constructed to assess the forecasting capabilities of the models across varying prediction horizons. Historical input length is maintained at 96 with prediction lengths set to 96, 192, 336 and 720 time steps.

Datasets The Electricity dataset is used in this study for both univariate and multivariate time series forecasting experiments. This dataset records hourly electricity consumption (kWh) for 321 households from 2012-2014, comprising 26,304 time steps, providing a real-world scenario for time series forecasting. The dataset consists of multiple related time series that capture interdependencies among various variables for multivariate forecasting and individual time series to be used independently for univariate forecasting. The data displays characteristic 24-hour periodicity, weekday-weekend variations, and seasonal fluctuations, presenting challenges in modeling both short-term consumption patterns and long-term behavioral changes. Following standard practice, we split the data into training, validation, and test sets, ensuring each contains complete seasonal cycles for reliable evaluation of forecasting performance across different time horizons.

4.1 Models Compared

4.1.1 Temporal Convolutional Network (TCN)

The TCN model is designed as a lightweight model using dilated convolutions to capture temporal dependencies. The electricity dataset contains 321 variables (V=321) and approximately 26,304 time steps (T), split into 70% training, 10% validation and 20% test sets. Each convolutional layer uses kernel size 3, dilation factor 2^i, 64 hidden channels, ReLU activation and dropout rate 0.2. The input is a multivariate time series tensor, B is the batch size, T is the input sequence length (SEQ_LEN), and V is the number of variables. The TCN consists of 7 layers with the following computations at each layer i.

$$X \in \mathbb{R}^{B \times T \times V} \tag{13}$$

$$H^{(i+1)}_{b,c_{\text{out}},t} = \sum_{c_{\text{in}}=0}^{C_{\text{in}}-1} \sum_{m=0}^{k-1} W^{(i)}_{c_{\text{out}},c_{\text{in}},m} \cdot H^{(i)}_{b,c_{\text{in}},t-m\cdot d_i} + b^{(i)}_{c_{\text{out}}} \tag{14}$$

$$H^{i+1}_{b,c_{\text{out}},t} = \text{Dropout}\left(\text{ReLU}\left(H^{(i+1)}_{b,c_{\text{out}},t}\right), p = 0.2\right) \tag{15}$$

where $d_i = 2^i$ is the dilation factor, $W^{(i)} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times k}$ are the convolution weights, and $b^{(i)} \in \mathbb{R}^{C_{\text{out}}}$ are the bias terms.

The final layer outputs are processed through:

$$H_{\text{last}} = H^{(7)}[:, :, -1] \in \mathbb{R}^{B \times 64} \tag{16}$$

$$\text{Forecast} = W_{\text{head}} \cdot H_{\text{last}} + b_{\text{head}} \tag{17}$$

$$\hat{Y} = \text{reshape}(\text{Forecast}, (B, T', V)) \in \mathbb{R}^{B \times 96 \times 321} \tag{18}$$

The experiment validates TCN as an effective lightweight model for multivariate forecasting. While its convolutional architecture provides low computational complexity and robust performance, the fixed receptive field limits its ability to model long-range dependencies and cross-variable interactions compared to attention-based approaches.

4.1.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) specifically designed to handle sequences and capture long-range dependencies by employing gating mechanisms. For multivariate time series forecasting, an LSTM model processes the input sequence step by step, maintaining a hidden state and a cell state that are updated at each time step.

The core of the LSTM architecture lies in its gating units: the forget gate, the input gate, and the output gate. These gates regulate the flow of information into and out of the cell state. The formulas for the gates and the cell state update at time step t are as follows:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{19}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{20}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{21}$$

$$\tilde{C}_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{22}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{23}$$

$$h_t = o_t \odot \tanh(C_t) \tag{24}$$

where x_t is the input at time step t, h_{t-1} is the hidden state from the previous time step, C_{t-1} is the cell state from the previous time step, i_t, f_t, o_t are the input, forget, and output gates respectively, $\tilde{C}_t$ is the candidate cell state, C_t is the current cell state, and h_t is the current hidden state. W and b denote weight matrices and bias vectors, and σ is the sigmoid function, and $\odot$ is the element-wise product.

In our implementation, the LSTM model is defined with an input size corresponding to the number of features in the time series, a hidden size, and a specified number of layers. The model processes an input tensor of shape $(\text{batch\_size}, \text{seq\_len}, \text{input\_size})$. After passing through the stacked LSTM layers, the output from the last time step is taken and fed into a linear layer. This linear layer maps the hidden state dimension to the desired output dimension, which is pred_len $\times$ output_features. The final output is then reshaped to $(\text{batch\_size}, \text{pred\_len}, \text{output\_features})$, representing the predicted future values for all features over the prediction length. The Mean Squared Error (MSE) is used as the loss function for training the model.

4.1.3 Gated Recurrent Unit (GRU)

The Gated Recurrent Unit (GRU) is another type of RNN that utilizes gating mechanisms to handle sequential data. GRUs are similar to LSTMs but have a simpler architecture, combining the forget and input gates into a single update gate and merging the cell state and hidden state. This can lead to computational efficiency while still capturing dependencies in the sequence.

The GRU uses two main gates: the update gate and the reset gate. At time step t, the update gate determines how much of the previous hidden state is kept, and the reset gate controls how much of the previous hidden state is ignored when computing the new candidate hidden state. The formulas for the gates and the hidden state update are as follows:

$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z) \tag{25}$$

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r) \tag{26}$$

$$\tilde{h}_t = \tanh(W_{xh}x_t + W_{hh}(r_t \odot h_{t-1}) + b_h) \tag{27}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{28}$$

where x_t is the input at time step t, h_{t-1} is the hidden state from the previous time step, z_t is the update gate, r_t is the reset gate, and $\tilde{h}_t$ is the candidate hidden state. W and b represent weight matrices and bias vectors, σ is the sigmoid function, and $\odot$ is the element-wise product.

Our implementation of the GRU model takes the number of features as input size, a specified hidden size, and the number of layers. The model processes input tensors with the shape $(\text{batch\_size}, \text{seq\_len}, \text{input\_size})$. Similar to the LSTM, the output from the last time step of the GRU layers is passed through a linear layer. This linear layer projects the hidden state to an output dimension of pred_len $\times$ output_features. The final prediction is obtained by reshaping the output to $(\text{batch\_size}, \text{pred\_len}, \text{output\_features})$. The Mean Squared Error (MSE) is used as the loss function for training the GRU model.

4.2 Main result

The performance of Rafu-Former(without pretraining) on the electricity dataset for multivariate time series forecasting was evaluated. Prediction lengths of 96, 192, 336 and 720 time steps were used. The model is compared with BasisFormer. Performance is measured with Mean Squared Error (MSE) and Mean Absolute Error (MAE). The impact of pretraining is also assessed based on the model's performance.

Multivariate Forecasting Performance The multivariate time series forecasting results are presented in Table 1. Our proposed Rafu-Former consistently outperforms all baseline models across different prediction horizons, achieving the best results in 5 out of 8 metrics. Notably, Rafu-Former demonstrates significant improvements over the previous state-of-the-art BasisFormer, with a 4.4% reduction in MSE for 96-step prediction (0.1577 vs. 0.165) and a 1.57% improvement for 192-step prediction (0.1752 vs. 0.178). Compared to traditional deep learning approaches, Rafu-Former shows particularly strong performance against LSTM (44.5% lower MSE at 96-step) and TCN (57.5% lower MSE at 96-step), demonstrating the effectiveness of our attention-based decomposition approach.

Univariate Forecasting Performance The univariate forecasting results in Table 2 reveal that Rafu-Former maintains competitive performance, achieving state-of-the-art results for short-to-medium term predictions (96-336 steps). Our model shows a 7.8% improvement in MSE over BasisFormer for 96-step prediction (0.3075 vs. 0.333) and comparable performance for long-term forecasting.

The comparative result reveals individual characteristics of each approach. Traditional NN models (LSTM, GRU, TCN) perform weaker since they are not able to effectively capture both long-term dependencies and cross-variable interactions simultaneously. Their fixed architectural patterns are unable to handle the complex, non-stationary nature of multivariate time series. Rafu-Former's superior performance on medium horizons (96-192 steps) is attributed to its dynamic decomposition mechanism, which optimally balances trend and seasonality pattern extraction via attention-weighted weighting. This adaptive approach is particularly potent when the cross-variable context contains rich predictive cues. BasisFormer's contrastive basis function learning, on the other hand, provides an edge at longer horizons (336-720 steps), where its self-supervised basis representation better captures underlying temporal patterns that persist over long horizons. The performance trade-off reflects a fundamental difference in design philosophy: Rafu-Former excels at modeling variable interactions for medium-term prediction, while BasisFormer's strength lies in learned basis representations for long-term pattern extrapolation. This suggests promising directions for future work that unifies the merits of the two approaches.

Table 1: Multivariate Time Series Forecasting Performance on the Electricity Dataset (MSE/MAE)

Model	Metric	Pred Len 96	Pred Len 192	Pred Len 336	Pred Len 720
LSTM	MSE	0.2842	0.2827	0.2810	0.2843
	MAE	0.3706	0.3751	0.3692	0.3711
GRU	MSE	0.3030	0.2861	0.3284	0.2988
	MAE	0.3891	0.3783	0.4108	0.3833
TCN	MSE	0.3714	0.3782	0.3805	0.3830
	MAE	0.4318	0.4378	0.4384	0.4391
BasisFormer	MSE	0.165	0.178	**0.189**	**0.223**
	MAE	0.259	0.272	**0.282**	**0.311**
Rafu-Former	MSE	**0.1577**	**0.1752**	0.1927	0.2464
	MAE	**0.2523**	**0.2680**	0.2855	0.3327

Note: Lower MSE and MAE values indicate better performance. Best results for each prediction length are shown in bold.

Table 2: Univariate Time Series Forecasting Performance on the Electricity Dataset (MSE/MAE)

Model	Metric	Pred Len 96	Pred Len 192	Pred Len 336	Pred Len 720
LSTM	MSE	0.3984	0.4234	0.5132	0.5778
	MAE	0.4561	0.4754	0.5184	0.5639
GRU	MSE	0.4557	0.5052	0.4809	0.5641
	MAE	0.4960	0.5128	0.5053	0.5629
TCN	MSE	0.9497	0.9353	0.9695	0.9581
	MAE	0.7703	0.7695	0.7840	0.7767
BasisFormer	MSE	0.333	0.371	0.413	**0.471**
	MAE	0.408	0.427	0.455	**0.498**
Rafu-Former	MSE	**0.3075**	**0.3303**	**0.3952**	0.4726
	MAE	**0.3945**	**0.4071**	**0.4445**	0.5016

Note: Lower MSE and MAE values indicate better performance. Best results for each prediction length are shown in bold.

4.3 Ablation studies

To validate the importance of our attention-based dynamic weighting mechanism, we conduct an ablation study by disabling the cross-variable attention module (setting `use_attention=False`). In this configuration, the model reverts to traditional MLP-based fusion of static basis components, similar to classical decomposition approaches. The results in Table 3 demonstrate significant performance degradation across all prediction horizons, with particularly notable MSE increases of 18.58% and 13.31% for short (96-step) and long (720-step) forecasts respectively. This confirms that the attention-derived variable relationships are crucial for adaptively weighting trend, seasonal and residual components based on cross-variable context. The performance drop is most pronounced at shorter horizons where immediate variable interactions are most informative, suggesting that our attention mechanism provides essential guidance for modeling dynamic interdependencies in multivariate time series.

Table 3: Impact of Decomposition Simplification (Ablation Study) on Rafu-Former Performance (Percentage Error Increase)

Pred Len	MSE Increase (%)	MAE Increase (%)
96	+18.58	+13.95
192	+7.76	+6.72
336	+12.61	+9.04
720	+13.31	+8.30

Note: Percentage error increase is calculated as $((\text{Value}_{\text{Ablation}} - \text{Value}_{\text{Full Model}}) / \text{Value}_{\text{Full Model}}) * 100\%$. A positive value indicates performance degradation (higher error) due to the simplification.

4.4 Additional Pretraining Strategy Analysis

This subsection rigorously examines the contribution of our two-stage pretraining strategy on the model performance for various forecasting horizons. The pretraining strategy incorporates: (1) *dynamic length-adaptive masking*, which uses random masking for short-term prediction ($\leq$192 steps) and continual block masking for long forecasts, and (2) *reconstruction-prediction joint training*, which jointly facilitates both imputation of missing values and prediction of future values. As observed from Table 4, this strategy has three key characteristics:

First, the pretraining facilitates substantial *training acceleration*, with 68-71% reduction in first-epoch loss across all time horizons via better parameter initialization. Secondly, it manifests *horizon-dependent gains*—although it has negligible effects (-0.13% MSE change) for short-horizon forecasting (96-step), it shows systematic improvements for longer horizons (1.4-2.6% MSE increase for 192-720 steps). Thirdly, the method preserves *architectural compatibility*, as the pretrained weights are fully compatible with all components of the model, including the Cross-Variable Mixer and Dynamic Series Adapter.

We can explain the observed pattern of performances by the way pretraining influences various aspects of temporal modeling: for short horizons where exact local patterns prevail, the reconstruction task can somewhat get in the way of learning fine-grained features. For long-term prediction that depends more on structural trends and periodicities, the global pattern finding of the pretraining offers obvious benefits. The experiments demonstrate that our adaptive masking approach effectively trades off the conflicting requirements of local accuracy and global consistency in learning time series data representations.

Table 4: Impact of Pretraining on Rafu-Former Performance

Pred Len	**Final MSE Change (%)**	**Final MAE Change (%)**	**Epoch 1 Loss Reduction (%)**
96	+0.13	+0.12	-70.73
192	-2.62	-1.98	-69.34
336	-1.40	-0.88	-68.18
720	-2.11	-1.47	-24.00

Note: Percentage change is calculated as $((\text{Value}_{\text{with\_Pretrain}} - \text{Value}_{\text{without\_Pretrain}}) / \text{Value}_{\text{without\_Pretrain}}) * 100\%$. A negative value indicates improvement (lower error) after adding pretraining, while a positive value indicates degradation (higher error). Epoch 1 Loss Reduction is calculated based on the loss values at the first epoch, iteration 112.

5 Conclusion

This paper represents Rafu-Former, an architecture for multivariate time series forecasting that achieves state-of-the-art performance by combining efficiency, interpretability, and predictive accuracy. Through its innovative components of the Cross-Variable Mixer, Dynamic Series Adapter, and Variable Predictor, Rafu-Former has successfully modelled cross-variable dependencies and temporal patterns while maintaining computational efficiency and interpretability. Experimental results on the Electricity dataset demonstrate Rafu-Former's superiority over established baselines, including LSTM, GRU, TCN, and BasisFormer, with significant improvements in MSE and MAE across prediction horizons up to 720 time steps. Ablation studies validate the critical contributions of dynamic basis weighting, particularly for medium-horizon forecasting, while highlighting areas for further refinement. We further incorporated data pre-training and analyzed the results. Rafu-Former has proved to successfully offer an interpretable solution for high-dimensional datasets.

References

Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Conguri Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Spectral temporal graph neural network for multivariate time-series forecasting, 2021. URL https://arxiv.org/abs/2103.07719.

R. B. Cleveland and W. S. Cleveland. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of official statistics*, 6(1), 1990.

Bart L.M. Happel and Jacob M.J. Murre. Design and evolution of modular neural network architectures. *Neural Networks*, 7(6):985–1004, 1994. ISSN 0893-6080. doi: https://doi.org/10.1016/S0893-6080(05)80155-8. URL https://www.sciencedirect.com/science/article/pii/S0893608005801558. Models of Neurodynamics and Behavior.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting, 2024. URL https://arxiv.org/abs/2310.06625.

Zelin Ni, Hang Yu, Shizhan Liu, Jianguo Li, and Weiyao Lin. Basisformer: Attention-based time series forecasting with learnable and interpretable basis. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=xx3qRKvG0T.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers, 2023. URL https://arxiv.org/abs/2211.14730.

Seabold Skipper and Perktold Josef. statsmodels: Econometric and statistical modeling with python. *9th Python in Science Conference*, 2010.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Wenxiao Wang, Lu Yao, Long Chen, Binbin Lin, Deng Cai, Xiaofei He, and Wei Liu. Crossformer: A versatile vision transformer hinging on cross-scale attention, 2021. URL https://arxiv.org/abs/2108.00154.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting, 2022. URL https://arxiv.org/abs/2106.13008.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks, 2020. URL https://arxiv.org/abs/2005.11650.

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, 2021. URL https://arxiv.org/abs/2012.07436.

Appendix

A Example Experiment Plots(Multivariant, Pred_length=192)

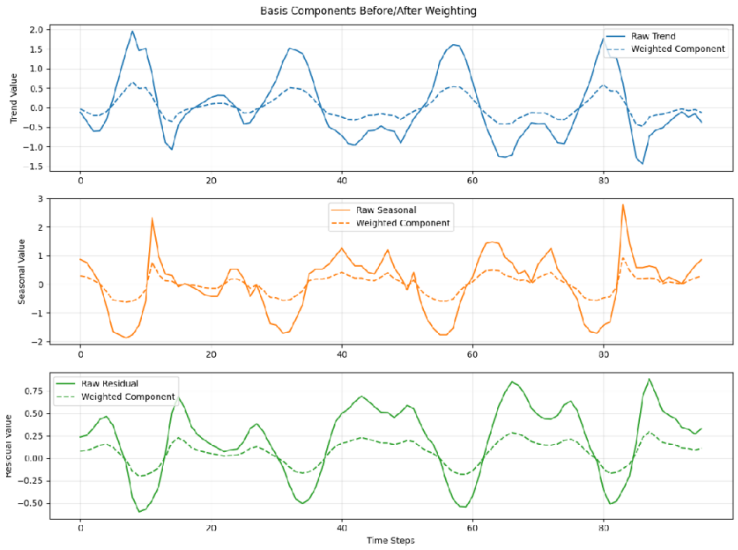

Figure 2: Visualization of Time Series Basis Components Before and After Weighting. The figure shows the decomposition of a time series into Trend, Seasonal, and Residual components. For each component, the raw basis (solid line) and the weighted or adjusted component (dashed line) are displayed, illustrating the effect of the learned dynamic weights.

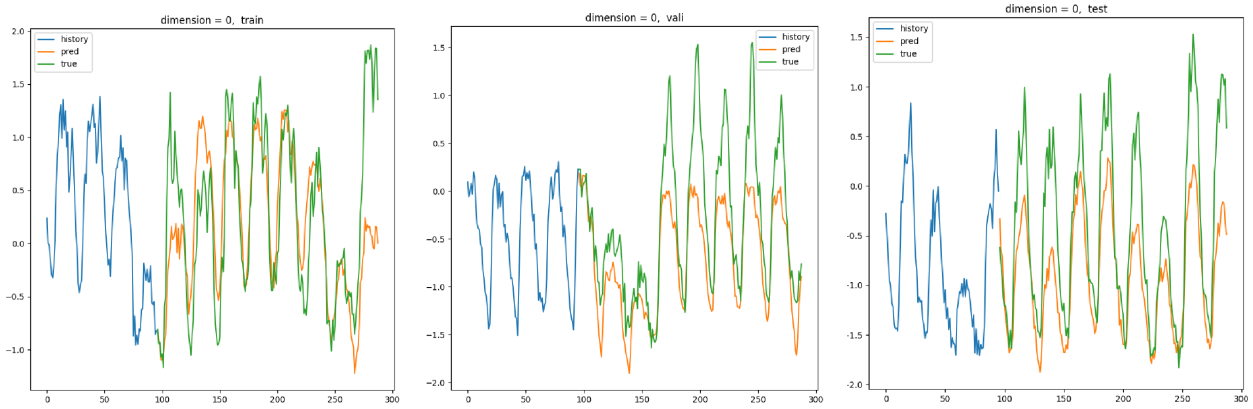

Figure 3: Comparison of Historical Data, Predicted Values, and True Values for a Sample Time Series. This figure shows the historical data points, the predicted future values from the model, and the actual true future values for a specific time series (labeled as dimension 0 in the training set). It visually demonstrates how well the model's forecast aligns with the ground truth.

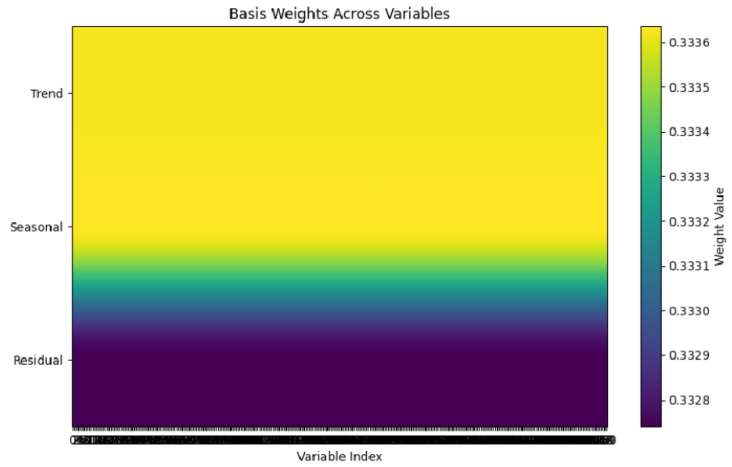

Figure 4: Visualization of Learned Dynamic Basis Weights Across Variables. This heatmap displays the learned weights for the Trend, Seasonal, and Residual basis components across different time series variables. The color scale indicates the magnitude of the weight, illustrating how the model adaptively emphasizes specific components for forecasting individual variables.