

Learning Temporal Saliency for Time Series Forecasting with Cross-Scale Attention

Ibrahim Delibasoglu^{a,b,*}, Fredrik Heintz^b^a*Department of Software Engineering Faculty of Computer and Information Sciences Sakarya University Sakarya, Turkiye*^b*Department of Computer and Information Science (IDA) REAL AIICS Linkoping University Linkoping, Sweden*

ARTICLE INFO

Keywords:

Time series
 Time series forecasting
 Explainability
 Temporal saliency

ABSTRACT

Explainability in time series forecasting is essential for improving model transparency and supporting informed decision-making. In this work, we present CrossScaleNet, an innovative architecture that combines a patch-based cross-attention mechanism with multi-scale processing to achieve both high performance and enhanced temporal explainability. By embedding attention mechanisms into the training process, our model provides intrinsic explainability for temporal saliency, making its decision-making process more transparent. Traditional post-hoc methods for temporal saliency detection are computationally expensive, particularly when compared to feature importance detection. While ablation techniques may suffice for datasets with fewer features, identifying temporal saliency poses greater challenges due to its complexity. We validate CrossScaleNet on synthetic datasets with known saliency ground truth and on established public benchmarks, demonstrating the robustness of our method in identifying temporal saliency. Experiments on real-world datasets for forecasting task show that our approach consistently outperforms most transformer-based models, offering better explainability without sacrificing predictive accuracy. Our evaluations demonstrate superior performance in both temporal saliency detection and forecasting accuracy. Moreover, we highlight that existing models claiming explainability often fail to maintain strong performance on standard benchmarks. CrossScaleNet addresses this gap, offering a balanced approach that captures temporal saliency effectively while delivering state-of-the-art forecasting performance across datasets of varying complexity.

1. Introduction

Time series analysis is a growing area of research with applications in forecasting, classification, imputation, and anomaly detection. Despite advancements, many industry applications require transparent and interpretable models, especially in high-stakes domains. This has driven the demand for Explainable Artificial Intelligence (XAI) in time series forecasting, where understanding the decision-making process is critical. While XAI has seen significant development in image-based tasks, its application to time series data remains less explored. Most XAI efforts in time series have focused on classification, with several reviews discussing interpretability techniques [1, 2, 3]. However, there is a noticeable gap in XAI for time series forecasting, which requires explaining both feature importance and temporal dependencies.

XAI methods can be categorized as post-hoc vs. antehoc, global vs. local, and model-agnostic vs. model-specific. Post-hoc methods, such as SHAP [4] and LIME [5], provide explanations after training and are widely used in regression and classification tasks. However, time series data involves temporal dependencies, requiring explanations that capture sequential relationships. Model-specific methods, such as N-BEATS [6] and Multilevel Wavelet Decomposition Networks [7], embed interpretability directly into the model. Attention mechanisms, as used in Temporal Fusion Transformer (TFT) [8] and Spatio-Temporal Attention LSTM [9],

have also been explored for interpretability, though their effectiveness is debated [10, 11].

Recent architectures in time series forecasting, such as NHITS [12], iTransformer [13], TimeMixer [14] and LMSAutoTSF [15], prioritize accuracy over interpretability. PatchTST [16] segments time series into patches for multivariate forecasting, while LMSAutoTSF uses multi-scale processing with autocorrelation for temporal processing in the encoders. ETSFormer [17] is a time-series Transformer architecture that modifies self-attention with exponential smoothing and frequency attention, enabling interpretable decomposition of time-series data into components like level, growth, and seasonality while improving forecasting accuracy and efficiency, making it a suitable choice for comparison in our study. In this paper, we focus on improving explainability in time series forecasting while maintaining competitive accuracy. Fully MLP-based methods like LMSAutoTSF and TimeMixer both achieve state-of-the-art (SOTA) performance by leveraging trend and seasonal decomposition. LMSAutoTSF processes trend and seasonal components across multiple scales, using autocorrelation as a skip connection for temporal processing and combining predictions through simple summation. It also employs decomposition in frequency domain with learnable cutoff frequencies. In contrast, TimeMixer applies a fixed decomposition and uses a mixing strategy, applying a fine-to-coarse approach for seasonal components and a coarse-to-fine approach for trend components to refine details. While TimeMixer introduces a relatively complex refinement strategy, LMSAutoTSF achieves slightly better performance with a simpler approach. However, neither method provides inherent explainability. In contrast, transformer-based methods such as iTransformer and PatchTST exhibit

*Corresponding author

✉ ibrahimdelibasoglu@sakarya.edu.tr (I. Delibasoglu); fredrik.heintz@liu.se (F. Heintz)
 ORCID(s): 0000-0001-8119-2873 (I. Delibasoglu); 0000-0002-9595-2471 (F. Heintz)

competitive performance on public benchmarks and offer the possibility of explainability through attention map visualization. However, attention mechanisms may focus on inputs due to positional biases or other learned patterns, which might not accurately represent true saliency. Notably, prior works [10, 11] highlight the limitations of attention mechanisms in providing true explainability. To address this, our work introduces an effective approach aimed at enhancing the saliency-capturing capacity of attention mechanisms through the proposed methodology.

Our primary contribution is addressing this gap by focusing on explainability for time series analysis mostly required for industrial tasks. We compare the interpretability of these models on a synthetic dataset with known feature and time-step importance, and evaluate their accuracy on public benchmarks. Building on the effective trend and seasonal component analysis of LMSAutoTSF and TimeMixer, we adopt a similar multi-scale decomposition approach but aim to provide built-in explainability. Unlike LMSAutoTSF and TimeMixer, which do not incorporate any intrinsic explainability mechanisms, our proposed solution enhances explainability while retaining competitive performance. Our architecture, CrossScaleNet, introduces an advanced patch-based cross-attention mechanism to improve interpretability without sacrificing accuracy. By incorporating cross-attention between scales, we enhance the ability to capture temporal dependencies and provide insights into time-step importance. We evaluate CrossScaleNet alongside LMSAutoTSF, TimeMixer, iTransformer, TFT, and PatchTST, demonstrating that our approach delivers superior explainability and competitive accuracy.

The primary contributions of this work are as follows:

- Development of a synthetic dataset to evaluate explainability methods in time series forecasting, focusing on temporal and feature importance.
- Proposal of CrossScaleNet, an advanced architecture with a patch-based cross-attention mechanism for improved multi-scale processing and interpretability.
- The proposed CrossScaleNet effectively captures temporal saliency on synthetic datasets and emerges as the best method, offering explainability and higher accuracy across long-term, and short-term forecasting tasks.

2. Literature Review

XAI for Image Data. Grad-CAM [18] explains differentiable classifiers like CNNs by linking predictions to input image regions, making it useful for image classification and object detection. DeepLIFT [19] compares neuron activations to a reference, attributing importance to input features, while Integrated Gradients [20] integrates gradients along a path between baseline and actual inputs for reliable feature attribution. SmoothGrad [21] enhances gradient-based interpretability by averaging gradients over noisy input copies, improving sensitivity maps.

XAI for Time Series Classification. Time series data introduces temporal dependencies, requiring XAI methods to explain feature importance and their evolution over time. SHAP and LIME have been adapted for time series, while model-specific approaches like multilevel wavelet decomposition networks [7] and CNN-RNN hybrids [22] generate interpretable representations. LEFTIST [23] is a model-agnostic framework for time series classification, and ETSCM [24] uses attention to identify important ECG segments. LAXCAT [25] employs convolutional attention for multivariate time series classification. Attention mechanisms, popularized by [26], are widely used in time series classification [27, 28, 29]. [30] proposes a framework with quantitative metrics to evaluate post-hoc interpretability methods for time-series classification.

XAI for Time Series Forecasting. Time series forecasting has seen fewer XAI developments due to its complexity, requiring models to account for both feature importance and future trajectories. ShapTime [31] adapts SHAP for forecasting by incorporating temporal dependencies. Counterfactual explanations [32] explore how predictions change with altered inputs, while TFT uses attention to capture temporal dependencies and provide interpretability. [33] explores interpretable deep learning for industrial forecasting, emphasizing trust in black-box models. [34] enhances explainability through visual association rules, and ShapTime [31] extends SHAP for temporal dynamics, improving practical applicability. EDformer [35] evaluates post-hoc XAI approaches for time series forecasting using transformer-based methods. While it demonstrates efficiency in interpretability, its overall accuracy on public benchmarks falls short compared to the latest architectures.

Dataset	Important lags	Important Features	Target Noise level
SYN1	1-15	{0, 1}	0.01
SYN2	1-5, 9-10, 15-16, 18, 20, 25-26, 35-36, 50-52, 91-95	{0, 2}	0.05
SYN3	9-10, 15-16, 18, 20-25, 31, 34, 60-65	{1, 2}	0.08
SYN4	9-10, 15-16, 18-21, 41-42, 45-46	{1, 2}	0.10
SYN5	71-77	{1, 2}	0.06
SYN6	48-57	{0, 2}	0.05
SYN7	60, 62-69	{0, 1}	0.02
SYN8*	mixture of SYN5 and SYN6	{0, 1, 2}	0.11

Table 1: Characteristics of Synthetic Datasets

3. Synthetic Time Series Generation

To evaluate the performance of time series models and saliency explanations, eight synthetic datasets with different characteristics are generated using pre-defined temporal dependencies, feature importance, and corresponding saliency maps. Consider a dataset $\mathbf{X} \in \mathbb{R}^{n_{\text{samples}} \times n_{\text{features}}}$, where each feature X_i is categorized as either important or non-important. The target variable y is constructed using both the current values of important features and their lagged values.

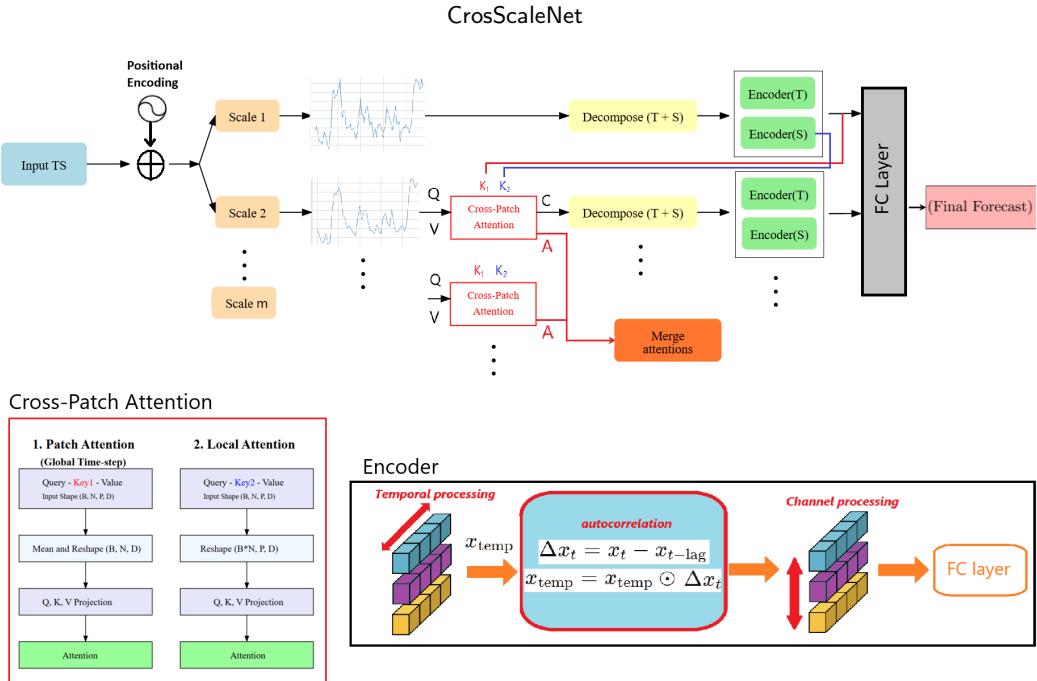


Figure 1: General overview of the multi-scale processing with patch-based cross-attention

Each dataset contains a unique combination of **Important Features** (indices of features with direct and lagged contributions to y), **Important Lags** (the set of lag values influencing the target variable), and **Noise Level** (standard deviation of Gaussian noise added to y). The saliency maps of datasets named *SYN1*, *SYN2*, *SYN3*, *SYN4*, *SYN5*, *SYN6*, *SYN7*, and *SYN8* exhibit distinct temporal focusing patterns, reflecting variability in the temporal importance across datasets as shared in Table 1. The first four datasets give more importance to recent historical points, while the others give more importance to the oldest historical points. This variability is crucial for evaluating the temporal saliency capability of models, particularly in assessing how well a model can identify and focus on the most relevant time steps for predictions.

4. Methodology

Given a D -dimensional multivariate time series $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T] \in \mathbb{R}^{D \times T}$ of sequence length T , where $\mathbf{x}_t \in \mathbb{R}^D$ represents the data at the t -th time step and D denotes the number of features, we aim to develop an explainable time series forecasting model. Our methodology offers intrinsic explainability by leveraging learned attention weights, enabling multi-scale processing that effectively captures and interprets the temporal importance of different time steps.

We introduce a patch-based cross-attention mechanism to capture distinct temporal dependencies, as shown in Figure 1. To ensure interaction across scales, cross-attention is applied between the highest-level merged and the seasonal component outputs. Our architecture combines multi-scale processing with efficient attention to capture temporal patterns at various granularities. The input series is decomposed into trend and seasonal components, processed

independently across scales using dedicated encoders. The outputs are merged for the final forecast, with attention maps interpolated and aggregated to provide a comprehensive view of temporal importance.

4.1. Multi-Scale Processing

The proposed model processes the input X at multiple scales $m \in \{1, \dots, M\}$. Scales except the highest one are refined using proposed cross-patch attention. Input data are decomposed into seasonal and trend components in each scale:

$$X_{\text{seasonal}}^m, X_{\text{trend}}^m = \text{Decomp}(X^m). \quad (1)$$

The decomposed components are processed through their respective Encoders including Fully Connected (*FC*) layers for both temporal and channel processing in the Encoder as detailed in Figure 1 to compute scale output y^m :

$$y^m = \text{Encoder}_{\text{Seasonal}}(X_{\text{seasonal}}^m) + \text{Encoder}_{\text{Trend}}(X_{\text{trend}}^m). \quad (2)$$

Each output is scaled using a learnable scale weight σ^m :

$$y^m = y^m \cdot \sigma^m, \quad \text{where } \sigma^m = \text{sigmoid}(w^m). \quad (3)$$

Finally, the predictions across all scales are combined to produce the final output y via last *FC* layer:

$$y = FC([y^1, y^2, \dots, y^m]) \quad (4)$$

4.1.1. Cross-attention via scales

For each scale, the model computes predictions and combines them to generate the final forecast via a *FC* layer. Notably, attention is only applied for scales $m > 1$, where the predictions from the first scale (y_1) are used as the keys for the attention module. For all subsequent scales ($m > 1$), the cross-patch attention mechanism refines the input window X^m by incorporating the predictions from the first scale. This prediction is appropriately scaled and interpolated to match the temporal resolution of X^m , updating the context (C) in alignment with the method described in section 4.2.

The attention A for scale $m > 1$ is computed as follows:

$$A^m = \text{CrossPatchAttention}(Q^m, K1, K2, V^m), \quad (5)$$

where:

- $Q^m = V^m = X^m$: The Query and Value matrices derived from the input at scale m .
- $K1 = y^1$: The key matrix derived by interpolating the first scale's prediction y^1 .
- $K2 = (y_{\text{seasonal}}^1)$: The key matrix derived by interpolating the first scale's seasonal encoder prediction y_{seasonal}^1 .

The attention mechanism is implemented using patch-based approach, which enhances computational efficiency and better captures temporal patterns by processing patches of the input sequence. Figure 3 illustrates the key difference between regular self-attention and the patch-based attention approach. Figure 2 illustrates the proposed Cross-Patch Attention mechanism, which integrates two complementary attention components:

- **Patch Attention:** Processes global temporal relationships by reshaping the input into the format (B, N, D) , where B is the batch size, N is the number of patches, and D is the feature dimension. This enables attention computation across patches, capturing long-range dependencies in the data.
- **Local Attention:** Focuses on fine-grained patterns within each patch by reshaping the input to $(B*N, P, D)$, where P is the patch length. This allows the model to capture detailed, localized information within individual patches.

The red and blue colored variables in Figure 2: **Key1** ($K1$) and **Key2** ($K2$), represent distinct inputs for the key component of the attention mechanism. These inputs play a critical role in computing attention scores, ensuring that both global and local patterns are effectively integrated. Together, these components enable the model to achieve a balance between capturing broad temporal trends and fine-grained details, enhancing its performance in time-series tasks.

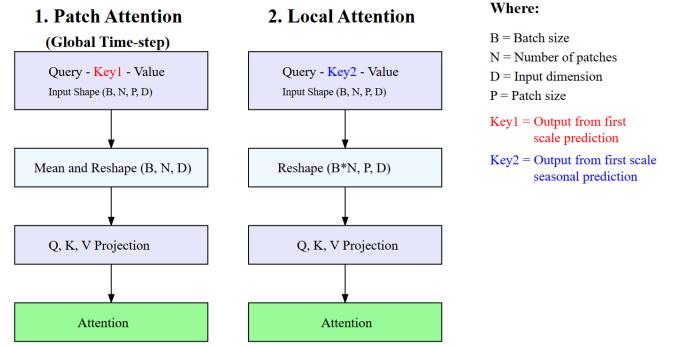


Figure 2: 1. Patch Attention processes global temporal relationships by reshaping input into (B, N, D) format, enabling attention computation across patches, 2. Local Attention captures fine-grained patterns by operating within each patch.

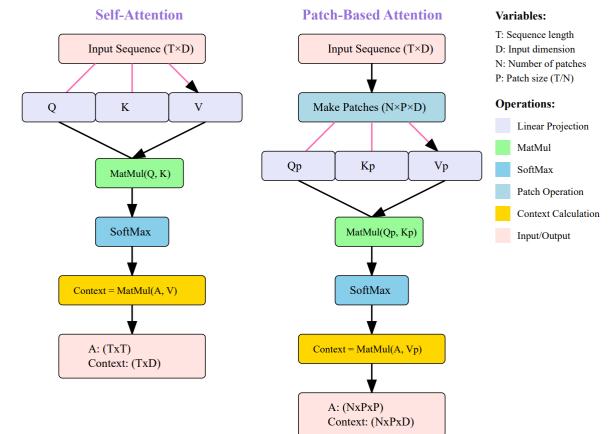


Figure 3: Comparison between Self-Attention and Patch-Based Attention mechanisms. Self-Attention (left) processes the entire sequence directly, projecting it into Query (Q), Key (K), and Value (V) representations. Patch-Based Attention (right) first segments the input into fixed-size patches before computing attention, enabling more efficient processing of long sequences.

4.2. Cross-Patch Attention Mechanism

Cross-patch attention mechanism is designed to handle hierarchical information. The base attention computation is defined as:

$$\text{Attention}(Q, K, V) = \text{Softmax} \left(\frac{QK^\top}{\sqrt{d}} \right) V \quad (6)$$

where Q, K, V are the query, key, and value representations, and d is a scaling factor to normalize attention scores. The mechanism employs two distinct attention modules: *Patch Attention* and *Local Attention*, each explained below.

4.2.1. Patch Attention

The *Patch Attention* mechanism captures relationships between patches across the entire sequence to model global dependencies. The input sequence is divided into patches:

$$X \rightarrow X_p \in \mathbb{R}^{B \times N \times P \times D} \quad (7)$$

where B is the batch size, N is the number of patches, P is the patch size, and D is the input dimension.

To compute patch-level attention, mean pooling is applied over the patch dimension to extract patch-level representations for the query (Pr_q), key (Pr_k), and value (Pr_v):

$$\text{Pr}_q = \frac{1}{P} \sum_{i=1}^P X_{p,q}^{(i)}, \quad \text{Pr}_k = \frac{1}{P} \sum_{i=1}^P X_{p,k}^{(i)}, \quad \text{Pr}_v = \frac{1}{P} \sum_{i=1}^P X_{p,v}^{(i)} \quad (8)$$

The **key** representation for Patch Attention ($X_{p,k}$) is derived from the *prediction output from the highest-scale y^1* , representing the prediction from seasonal and trend encoders for highest scale. This design choice ensures that the global dependencies across patches are captured using the most comprehensive temporal representation. The attention computation proceeds as follows:

$$Q_p = W_q \text{Pr}_q, \quad K_p = W_k \text{Pr}_k, \quad V_p = W_v \text{Pr}_v, \quad (9)$$

where $W_q, W_k, W_v \in \mathbb{R}^{D \times D}$ are learnable weight matrices. The patch-level attention (A_p) is computed as:

$$A_p = \text{Softmax} \left(\frac{Q_p K_p^\top}{\sqrt{D}} \right) V_p \quad (10)$$

The output of Patch Attention represents the aggregated global context information across patches.

4.2.2. Local Attention

The *Local Attention* mechanism focuses on capturing fine-grained relationships within each patch. Each patch representation (X_p) is reshaped for local attention computation as:

$$X_L \in \mathbb{R}^{(B \times N) \times P \times D} \quad (11)$$

The **key** representation for Local Attention ($X_{l,k}$) is derived from the *prediction output from the highest-scale seasonal component y_{season}^1* , representing the prediction from seasonal encoder for highest scale which captures the periodic patterns within the sequence. By using this specialized representation, the Local Attention mechanism emphasizes the relationships between time steps that contribute to seasonal variations. The attention computation is defined as:

$$Q_L = W_{lq} X_L, \quad K_L = W_{lk} X_L, \quad V_L = W_{lv} X_L \quad (12)$$

$$A_L = \text{Softmax} \left(\frac{Q_L K_L^\top}{\sqrt{D}} \right) V_L \quad (13)$$

The result of Local Attention captures the within-patch context, allowing the model to resolve temporal relationships specific to short-term dependencies.

4.2.3. Cross-Patch Attention Context Fusion

The outputs of the Patch Attention and Local Attention mechanisms are combined to generate the final representation. The patch-level context from the Patch Attention mechanism is computed as:

$$C_P = A_p \cdot V_p, \quad \text{where } C_P \in \mathbb{R}^{B \times N \times D} \quad (14)$$

To align with the temporal resolution of the sequence, C_P is reshaped and broadcasted across the time steps within each patch:

$$C_P \rightarrow C_P \in \mathbb{R}^{B \times N \times P \times D} \quad (15)$$

Similarly, the local context from the Local Attention mechanism is defined as:

$$C_L = A_L \cdot V_L, \quad \text{where } C_L \in \mathbb{R}^{B \times N \times P \times D} \quad (16)$$

The final output of the Cross-Patch Attention output (\mathbf{C}) is obtained by combining the broadcasted patch context and the local context:

$$\mathbf{C} = C_P + C_L \quad (17)$$

4.3. Theoretical Motivation for Cross-Scale Attention and Interpretability

The cross-scale attention mechanism in our model enhances interpretability by aligning hierarchical temporal abstractions with attention-driven relevance signals. Unlike conventional self-attention which compares positions within the same input scale, our approach introduces a semantic alignment across scales by comparing temporal patches at lower scales with abstracted predictions from higher scales. This design enables the model to focus on relevant patterns across resolutions, grounded in learned predictive structure rather than raw input signals.

Interpretability via Hierarchical Temporal Abstraction. Each scale m represents a different level of temporal granularity. Lower scales preserve fine-grained, high-frequency patterns, while higher scales encode coarse, long-range dynamics. Interpretability improves when the model can connect these low-level signals to high-level abstractions. Formally, we define a hierarchy of abstraction functions $h_m : \mathbb{R}^{D \times T} \rightarrow \mathbb{R}^{D \times T_m}$, where $T_m < T_{m-1}$, and each abstraction is associated with a corresponding prediction y^m . These hierarchical outputs serve as interpretable anchors for multi-resolution attention.

Category	Dataset	Dim	Frequency	Information
Long-term	ETTh1, ETTh2	7	Hourly	Electricity
	ETTm1, ETTm2	7	15 min	Electricity
	Weather	21	10 min	Weather
	Electricity	321	Hourly	Electricity
	Traffic	862	Hourly	Transportation
	Exchange	8	Daily	Economy
	Solar-Energy	137	10 min	Electricity
	Beijing PM2.5	12	Hourly	Meteorological
Short-term	PEMS03	358	5 min	Transportation
	PEMS04	307	5 min	Transportation
	PEMS07	883	5 min	Transportation
	PEMS08	170	5 min	Transportation

Table 2: Detailed dataset descriptions for forecasting datasets

Cross-Patch Attention as Semantic Matching. Our patch-based cross-attention compares localized temporal patches at scale m against semantic keys derived from predictions y^1 (and its seasonal component y_{seasonal}^1), rather than using the input itself as in self-attention. This key innovation grounds the attention mechanism in semantically meaningful patterns—e.g., trends or periodicities—rather than arbitrary correlations in the input. As a result, attention scores reflect meaningful temporal relevance, improving transparency of the model’s decision process. By comparing each patch to higher-level predictive contexts, the model learns to associate specific temporal regions with their abstract outcomes.

Information Bottleneck Interpretation. From an information theoretic perspective, this cross-scale conditioning acts as a form of relevance filtering via a soft information bottleneck. Let $I(X^m; y^m)$ denote the mutual information between the input at scale m and its forecast. Conditioning this mapping on the higher-level output y^1 , which serves as a compact, semantically-rich representation, yields a constrained dependency:

$$I(X^m; y^m | y^1) < I(X^m; y^m)$$

This inequality reflects a desirable regularization: only the components of X^m that align with the abstract, generalizable structure captured in y^1 contribute significantly to prediction. In practice, this filtering mechanism suppresses noisy or redundant attention patterns and sharpens focus on temporally salient regions, which leads to more interpretable and stable importance scores across scales.

5. Experimental results

We assess temporal saliency on synthetic datasets with known temporal importance, comparing CrossScaleNet to iTransformer, PatchTST and TFT. While TFT emphasizes interpretability and PatchTST uses patch-based processing, CrossScaleNet distinguishes itself through its multi-scale cross-attention mechanism. We visualize attention weights of PatchTST and iTransformer to analyze their temporal processing behavior. We also assess feature importance on the synthetic dataset using feature ablation and integrated gradients, and evaluate temporal saliency through comprehensiveness and sufficiency metrics. We evaluate forecasting

accuracy on real-world benchmark datasets for both short-term PEMS dataset [36] and long-term datasets used by Autoformer [37]. The Beijing PM2.5 Dataset [38] features multiple inputs and a single target, unlike others. We use it to analyze temporal saliency in multiple-input, single-target predictions.

5.1. Dataset Descriptions

Table 2 represents the details of datasets used in this study. The PEMS dataset [36], which is used for short-term traffic flow forecasting, includes four public traffic network datasets: PEMS03, PEMS04, PEMS07, and PEMS08. It provides an additional challenge with its highly dynamic and rapidly changing time series. We also evaluate the performance of our proposed architecture on eight widely-used public benchmark datasets, including Weather, Electricity, Traffic, Exchange and four ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2) for long-term forecasting used by Autoformer [37]. The Electricity Transformer Temperature (ETT) dataset contains 7 factors of electricity transformer measurements spanning with four subsets: ETTh1 and ETTh2 recorded hourly, and ETTm1 and ETTm2 recorded every 15 minutes. The Exchange dataset comprises daily exchange rates from 8 countries between 1990 and 2016. Additional datasets include Weather (21 meteorological factors collected every 10 minutes), Electricity Consumption Load (ECL) with hourly data from 321 clients, and Traffic data consisting of hourly road occupancy rates from 862 sensors in the San Francisco Bay area.

5.2. Ablation Studies

Table 3 presents an ablation study comparing the proposed architecture’s performance with two baseline attention mechanisms: self-attention and patch-based attention. In this study, Patch-cross^{key} refers to the proposed cross-attention mechanism, where the same key is shared for both patch and local attention. Meanwhile, Patch-cross^{key *} represents the variant where different keys are used for patch and local attention. Although Patch-cross^{key *} yields relatively lower performance for some synthetic datasets, our experiment focuses on temporal saliency. The results demonstrate that using the seasonal component output as a key for the local patch is more effective in capturing temporal saliency. The results clearly demonstrate the effectiveness of the proposed Cross-Patch Attention mechanism, which

CrosScaleNet

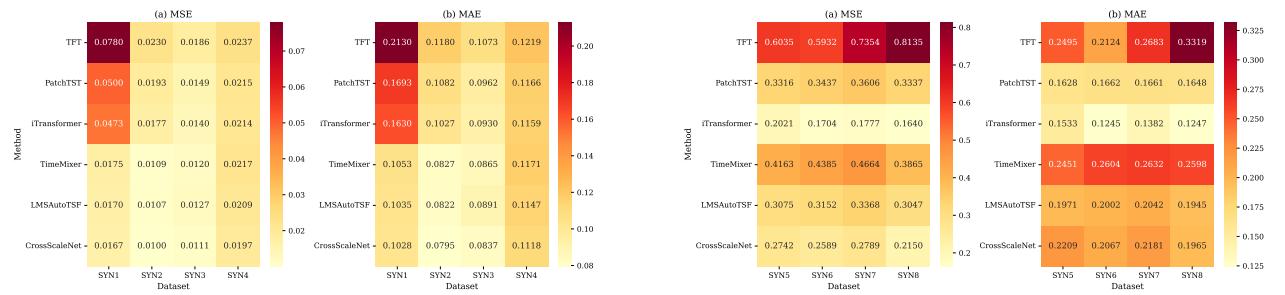


Figure 4: Performance comparison for forecasting accuracy on synthetic datasets.

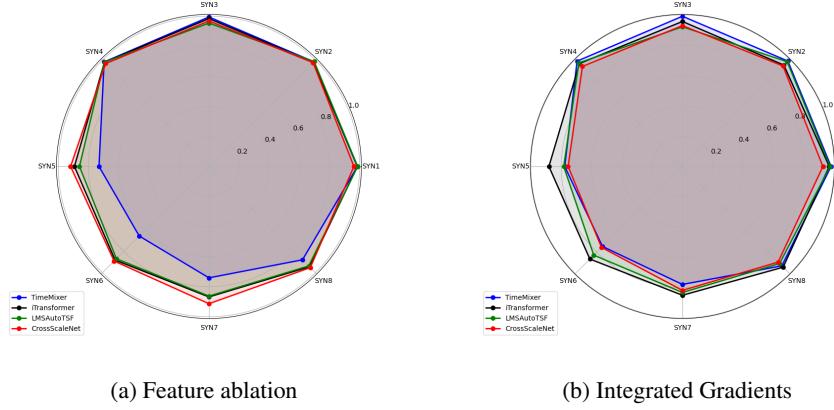


Figure 5: Comparison of important feature scores for each dataset with different methods via feature ablation.

combines multi-scale processing and cross-attention to enhance the modeling of temporal dependencies. Consistent improvements in performance are observed across multiple synthetic datasets (SYN1 - SYN8), as indicated by the evaluation metrics: Mean Squared Error (MSE) and Mean Absolute Error (MAE). Performance metrics (MSE, MAE) for the synthetic dataset are shown in Figure 4, alongside comparisons to TFT, PatchTST, iTransformer, TimeMixer, and LMSAutoTSF.

5.3. Comparative Analysis of Explainability Methods

Figure 8 and 9 illustrate the temporal saliency maps generated by the proposed CrossScaleNet, iTransformer, PatchTST, and TFT models. PatchTST’s attention weights roughly capture temporal saliency, focusing on both recent and distant time points. Since it processes each channel independently, the model uses only the target’s own history, without incorporating information from other features. TFT struggles particularly with SYN1 and SYN4, but it may capture temporal saliency relatively better for the other datasets. But its forecasting performance is not good enough as seen in the synthetic dataset and other real-world datasets. iTransformer can not capture temporal saliency on synthetic dataset as seen from the attention map visualization. On the other hand, proposed CrossScaleNet is able to capture temporal saliency in both the last and middle

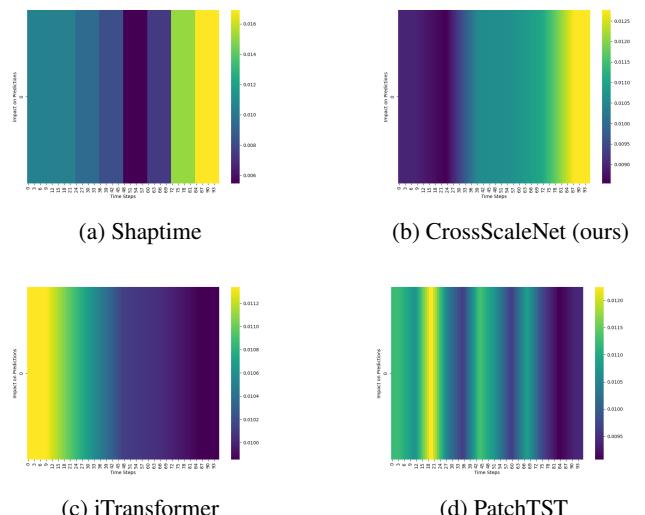


Figure 6: Temporal saliency for ETTh1 sample prediction

time steps, even if it does not always pinpoint the exact time steps perfectly. We can easily say that CrossScaleNet detects temporal saliency with higher sensitivity than the other models. Our experiments demonstrate that perfectly identifying the exact time step importance is challenging. Nevertheless, our solution offers a reasonable approximation

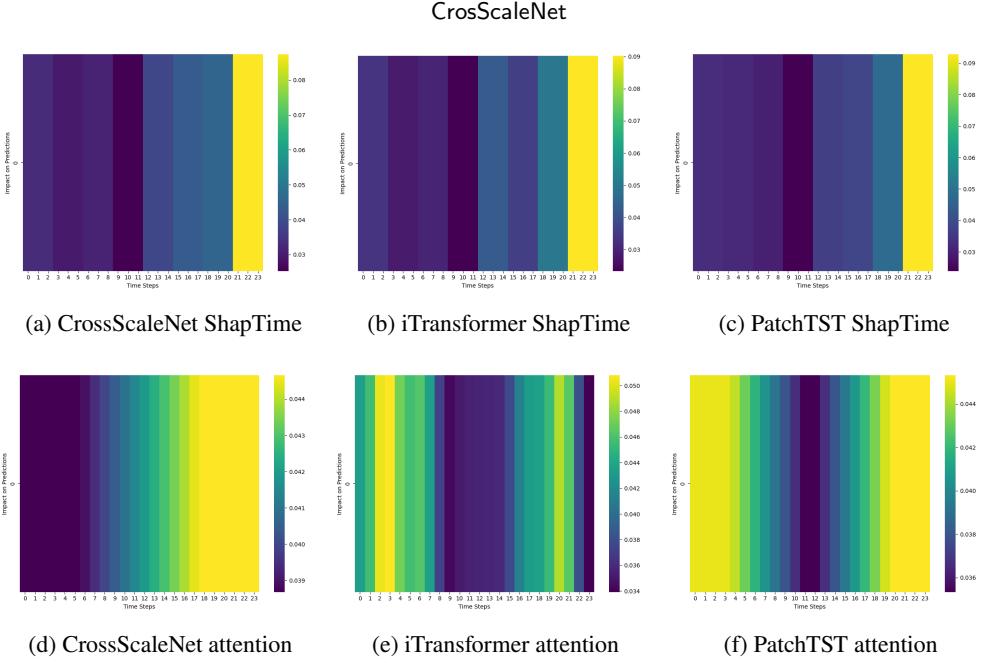


Figure 7: ShapTime and attention comparison on PM2.5 dataset.

of the temporal saliency with a competitive performance on public real-world benchmarks.

Figure 6 compares ShapTime with attention maps for sample prediction on real-world dataset ETTh1. ShapTime consistently provides reliable temporal saliency, but it is computationally heavy. It is model-agnostic, meaning it does not depend on the architecture or internal mechanisms of the model. Attention maps can be influenced by positional or structural biases inherent to the model, potentially leading to less accurate saliency interpretations. The figure shows that our method effectively captures temporal saliency by producing results closely aligned with ShapTime. We have also utilized *Beijing PM2.5* dataset for explainability comparisons by evaluating ShapTime and attention maps. The results, as shown in Figure 7, demonstrate that only CrossScaleNet produces outputs similar to ShapTime for PM2.5 dataset, highlighting the temporal saliency capacity of our approach. It is also important that each method gives same temporal saliency with ShapTime posthoc XAI method.

We analyze feature importance using Feature Ablation, and Integrated Gradients. Figure 5 shows feature importance scores from post-hoc explainability techniques. These scores demonstrate the ability of models to detect important features in the synthetic dataset. Our results show that our proposed model outperforms others, especially in detecting important features in the challenging SYN5-8 datasets with simple feature ablation, while iTransformer captures the important features better via Integrated Gradients. Our results demonstrate that the proposed model consistently outperforms others, particularly in identifying important features in the challenging SYN5-8 datasets through simple feature ablation. While our model excels at accurate feature detection, TimeMixer struggles with feature identification

on these datasets despite using similar multi-scale processing. Meanwhile, iTransformer effectively captures important features using Integrated Gradients; however, this approach incurs significantly higher computational costs compared to feature ablation.

Temporal saliency is quantitatively assessed using the *sufficiency* and *comprehensiveness* metrics on real-world datasets, with varying percentage thresholds applied to identify salient time steps. Table 4 reports the results for these two metrics across three time series models CrossScaleNet (ours), iTransformer, and PatchTST evaluated on two datasets: Electricity and PM2.5. All models extract temporal importance using an *intrinsic* approach via their respective attention mechanisms.

5.4. Performance Evaluation on Real-World Datasets

Table 5 presents the performance comparison for multivariate input with single-target prediction, where our method demonstrates the best overall performance while also offering explainability. Values highlighted in **red** and **blue** indicate the best and second-best results, respectively. The evaluation was conducted for long-term forecasting with a fixed look-back window of 96, as detailed in Table 6. Additionally, Table 7 compares multivariate short-term forecasting performance across different prediction horizons (12, 24, and 48), using the same look-back window.

6. Conclusion

Understanding which parts of a time series contribute most to a model’s predictions (temporal salience) is critical to improving transparency, confidence, and decision-making

Dataset	Model	MSE	MAE
SYN1	Self-Attention	0.3022	0.4686
	Patch-attention	0.0187	0.1077
	Patch-cross ^{key}	0.0167	0.1027
	Patch-cross ^{key} *	0.0167	0.1026
SYN2	Self-Attention	0.1290	0.3005
	Patch-attention	0.0152	0.0964
	Patch-cross ^{key}	0.0099	0.0792
	Patch-cross ^{key} *	0.0099	0.0793
SYN3	Self-Attention	0.0137	0.0921
	Patch-attention	0.0135	0.0916
	Patch-cross ^{key}	0.112	0.0837
	Patch-cross ^{key} *	0.111	0.0836
SYN4	Self-Attention	0.0196	0.1114
	Patch-attention	0.0315	0.1399
	Patch-cross ^{key}	0.0197	0.1118
	Patch-cross ^{key} *	0.0197	0.1118
SYN5	Self-Attention	0.3469	0.2441
	Patch-attention	0.2809	0.2314
	Patch-cross ^{key}	0.2572	0.2194
	Patch-cross ^{key} *	0.2742	0.2209
SYN6	Self-Attention	0.3451	0.2495
	Patch-attention	0.3048	0.2363
	Patch-cross ^{key}	0.2429	0.2043
	Patch-cross ^{key} *	0.2589	0.2067
SYN7	Self-Attention	0.3116	0.2357
	Patch-attention	0.3227	0.2513
	Patch-cross ^{key}	0.2618	0.2165
	Patch-cross ^{key} *	0.2788	0.2180
SYN8	Self-Attention	0.3308	0.2374
	Patch-attention	0.2612	0.2246
	Patch-cross ^{key}	0.1899	0.1888
	Patch-cross ^{key} *	0.2149	0.1964

Table 3: Ablation study for CrosScaleNet

in time series forecasts. However, attention-based models often learn positional or structural biases that do not accurately reflect true temporal importance. In this work, we address this challenge by proposing CrosScaleNet, a novel architecture that integrates cross-scale attention to more effectively capture meaningful temporal saliency. Our results on synthetic datasets with known saliency, along with evaluations on established public benchmarks and real-world forecasting tasks, confirm that CrosScaleNet consistently outperforms state-of-the-art transformer-based models. Importantly, it

achieves this while providing intrinsic interpretability, offering actionable insights without compromising predictive accuracy.

6.1. Model Complexity

Table 8 compares the computational requirements of CrosScaleNet against contemporary baselines. While iTransformer and PatchTST demonstrate superior training efficiency (2.012s and 2.024s per epoch respectively), CrosScaleNet achieves competitive performance (4.663s/epoch) with only 0.03G FLOPs matching LMSAutoTSF’s theoretical complexity while introducing multi-scale interpretability. Notably, CrosScaleNet maintains practical inference speeds (8.3ms/batch) comparable to TimeMixer (9.1ms), and significantly faster than TFT’s 112ms latency. The memory footprint remains stable across most models (336-344MB), with CrosScaleNet showing no significant overhead from its cross-attention mechanisms compared to the 380MB requirement of TFT.

7. Future work

This work has focused on modeling temporal saliency in multivariate time series using a unified attention mechanism. While our approach effectively captures temporal saliency and achieves strong predictive performance, an important aspect that could further enhance the modeling process is the differentiated influence of target and non-target features over time. In many forecasting tasks, recent target values may contribute more strongly to predictions than auxiliary features or long-range history. Exploring architectures that explicitly separate the processing of target and non-target features within the attention mechanism could provide even more fine-grained saliency estimation and potentially improve performance in scenarios where feature relevance varies significantly over time. This direction offers a promising opportunity to refine both the interpretability and accuracy of time series models across complex, real-world datasets.

8. Data availability

All the dataset are available online. The code to reproduce this work is available at <https://github.com/mribrahim/LMS-TSF>.

Dataset	Ratio	Sufficiency (Lower is better)			Comprehensiveness (Higher is better)		
		CrossScaleNet	iTransformer	PatchTST	CrossScaleNet	iTransformer	PatchTST
Electricity	10%	0.591	0.724	0.677	0.167	0.132	0.101
	20%	0.522	0.634	0.611	0.242	0.227	0.175
	50%	0.331	0.393	0.371	0.430	0.496	0.418
PM2.5	10%	0.484	0.674	0.55	0.373	0.076	0.139
	20%	0.449	0.664	0.537	0.366	0.104	0.157
	50%	0.389	0.656	0.203	0.432	0.175	0.519

Table 4: Comparison of Sufficiency and Comprehensiveness across Datasets

	CrossScaleNet		LMSAutoTSF		TimeMixer		iTransformer		PatchTST	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
12	0.3722	0.3789	0.3654	0.3783	0.3731	0.3791	0.4108	0.4024	0.4202	0.4093
24	0.6059	0.5068	0.5936	0.5053	0.6402	0.5254	0.6510	0.5294	0.6586	0.5275
48	0.9088	0.6518	0.9125	0.6557	0.9271	0.6623	0.9767	0.6751	0.9753	0.6728
Avg.	0.6289	0.5125	0.6238	0.5131	0.6468	0.5223	0.6795	0.5356	0.6847	0.5365

Table 5: Comparison of models on PM2.5 Dataset with multivariate input single target. (prediction horizons (12,24,48) and fixed look-back 24)

Models		CrossScaleNet	LMSAutoTSF	TimeMixer	iTransformer	PatchTST	ETSformer	TFT	
Database	Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.393	0.401	0.383	0.397	0.378	0.399	0.386	0.405
	192	0.445	0.429	0.435	0.426	0.440	0.431	0.443	0.434
	336	0.483	0.448	0.469	0.442	0.499	0.459	0.492	0.463
	720	0.490	0.473	0.478	0.464	0.549	0.510	0.508	0.491
	Avg.	0.453	0.438	0.441	0.432	0.466	0.449	0.457	0.448
ETTh2	96	0.296	0.344	0.293	0.344	0.292	0.343	0.294	0.346
	192	0.381	0.398	0.368	0.393	0.383	0.403	0.379	0.398
	336	0.413	0.427	0.411	0.427	0.439	0.437	0.434	0.435
	720	0.437	0.451	0.432	0.447	0.441	0.450	0.467	0.466
	Avg.	0.382	0.405	0.376	0.403	0.389	0.408	0.394	0.411
ETTm1	96	0.326	0.365	0.318	0.357	0.323	0.362	0.341	0.376
	192	0.378	0.394	0.357	0.378	0.368	0.386	0.381	0.395
	336	0.428	0.422	0.385	0.403	0.401	0.406	0.419	0.418
	720	0.484	0.452	0.450	0.441	0.431	0.441	0.486	0.455
	Avg.	0.404	0.408	0.377	0.394	0.380	0.398	0.406	0.411
ETTm2	96	0.174	0.258	0.171	0.254	0.175	0.256	0.184	0.266
	192	0.240	0.300	0.236	0.298	0.238	0.299	0.252	0.311
	336	0.299	0.339	0.295	0.337	0.299	0.339	0.313	0.349
	720	0.398	0.398	0.406	0.402	0.401	0.405	0.409	0.409
	Avg.	0.278	0.324	0.277	0.323	0.278	0.325	0.290	0.333
Weather	96	0.158	0.205	0.154	0.202	0.162	0.208	0.179	0.219
	192	0.209	0.252	0.202	0.247	0.207	0.251	0.225	0.258
	336	0.271	0.295	0.259	0.288	0.263	0.292	0.279	0.298
	720	0.356	0.351	0.338	0.340	0.343	0.344	0.328	0.351
	Avg.	0.249	0.276	0.238	0.269	0.244	0.274	0.253	0.282
Electricity	96	0.155	0.260	0.154	0.252	0.158	0.249	0.149	0.240
	192	0.171	0.272	0.165	0.260	0.169	0.259	0.164	0.255
	336	0.192	0.290	0.177	0.274	0.189	0.281	0.183	0.275
	720	0.217	0.307	0.204	0.299	0.228	0.314	0.228	0.312
	Avg.	0.184	0.282	0.175	0.271	0.186	0.276	0.181	0.270
Traffic	96	0.495	0.327	0.485	0.330	0.469	0.291	0.417	0.287
	192	0.495	0.327	0.483	0.322	0.480	0.299	0.432	0.292
	336	0.508	0.333	0.493	0.323	0.498	0.301	0.449	0.305
	720	0.545	0.352	0.524	0.336	0.506	0.313	0.481	0.318
	Avg.	0.511	0.335	0.496	0.328	0.488	0.301	0.445	0.301
Exchange	96	0.087	0.204	0.082	0.199	0.088	0.208	0.098	0.226
	192	0.180	0.301	0.174	0.296	0.184	0.307	0.188	0.312
	336	0.342	0.424	0.324	0.410	0.376	0.442	0.337	0.422
	720	0.877	0.703	0.834	0.686	0.880	0.701	0.885	0.714
	Avg.	0.372	0.408	0.354	0.398	0.382	0.415	0.377	0.419
Solar	96	0.214	0.264	0.214	0.264	0.259	0.293	0.228	0.271
	192	0.247	0.285	0.251	0.287	0.276	0.312	0.267	0.293
	336	0.261	0.298	0.270	0.299	0.296	0.326	0.286	0.310
	720	0.261	0.296	0.269	0.302	0.297	0.318	0.301	0.320
	Avg.	0.246	0.286	0.251	0.288	0.282	0.312	0.271	0.299
Overall Avg		0.342	0.351	0.332	0.345	0.344	0.351	0.341	0.353

Table 6: Comparison of multivariate long-term forecasting results with prediction horizons (96,192,336,720) and fixed look-back 96.

Models		CrossScaleNet		LMSAutoTSF		TimeMixer		iTTransformer		PatchTST	
Database	Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS3	12	0.0651	0.1699	0.0653	0.1703	0.0656	0.1720	0.0733	0.1803	0.0874	0.2028
	24	0.0889	0.1987	0.0877	0.1987	0.0921	0.2031	0.1062	0.2189	0.1344	0.2477
	48	0.1395	0.2479	0.1435	0.2564	0.1468	0.2610	0.1806	0.2911	0.2372	0.3339
	Avg	0.0978	0.2055	0.0988	0.2085	0.1015	0.2120	0.1200	0.2301	0.1530	0.2615
PEMS4	12	0.0728	0.1771	0.0741	0.1786	0.0737	0.1795	0.0939	0.2009	0.1062	0.2216
	24	0.0869	0.1960	0.0894	0.1981	0.0934	0.2081	0.1364	0.2459	0.1701	0.2817
	48	0.1154	0.2293	0.1236	0.2391	0.1232	0.2411	0.2291	0.3286	0.3124	0.3899
	Avg	0.0917	0.2008	0.0957	0.2053	0.0968	0.2096	0.1531	0.2585	0.1962	0.2977
PEMS7	12	0.0652	0.1647	0.0606	0.1580	0.0611	0.1582	0.0716	0.1744	0.0817	0.1953
	24	0.0835	0.1835	0.0837	0.1864	0.0836	0.1839	0.1092	0.2171	0.1353	0.2473
	48	0.1428	0.2594	0.1216	0.2269	0.1199	0.2219	0.1839	0.2881	0.2568	0.3452
	Avg	0.0972	0.2025	0.0886	0.1904	0.0882	0.1880	0.1216	0.2265	0.1579	0.2626
PEMS8	12	0.0788	0.1798	0.0775	0.1801	0.0769	0.1808	0.0980	0.2055	0.0990	0.2148
	24	0.1123	0.2140	0.1090	0.2136	0.1097	0.2165	0.1559	0.2646	0.1554	0.2649
	48	0.1793	0.2718	0.1757	0.2719	0.1949	0.2951	0.2922	0.3749	0.2757	0.3558
	Avg	0.1235	0.2219	0.1207	0.2219	0.1272	0.2308	0.1820	0.2817	0.1767	0.3118
Overall Avg		0.1025	0.2077	0.1010	0.2065	0.1034	0.2101	0.1442	0.2492	0.1710	0.2834

Table 7: Comparison of multivariate short-term forecasting results with different prediction horizons (12,24,48) and fixed look-back 96.

Models	FLOPs (G)	Training Time (seconds for 1 epoch)	Inference Latency (ms for one batch)	GPU Memory (MB)
CrossScaleNet	0.03	4.663	8.3	336
LMSAutoTSF	0.03	4.858	6.5	336
TimeMixer	0.13	5.341	9.1	344
iTTransformer	0.01	2.012	3.6	262
PatchTST	0.05	2.024	3.6	338
TFT	0.47	5.827	112	380

Table 8: Computational requirements comparison across models. Lower values are better for all metrics.

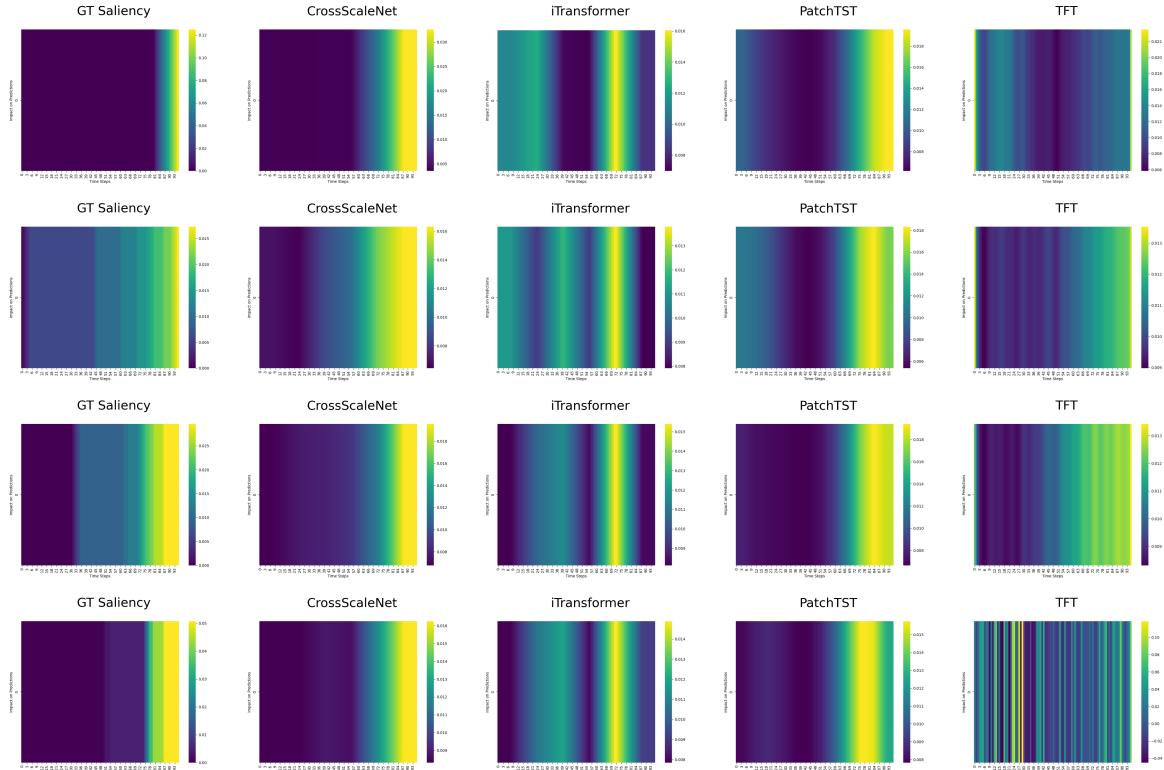


Figure 8: Visualization of ground truth and predicted saliency maps via different architectures for synthetic datasets: SYN1-2-3-4

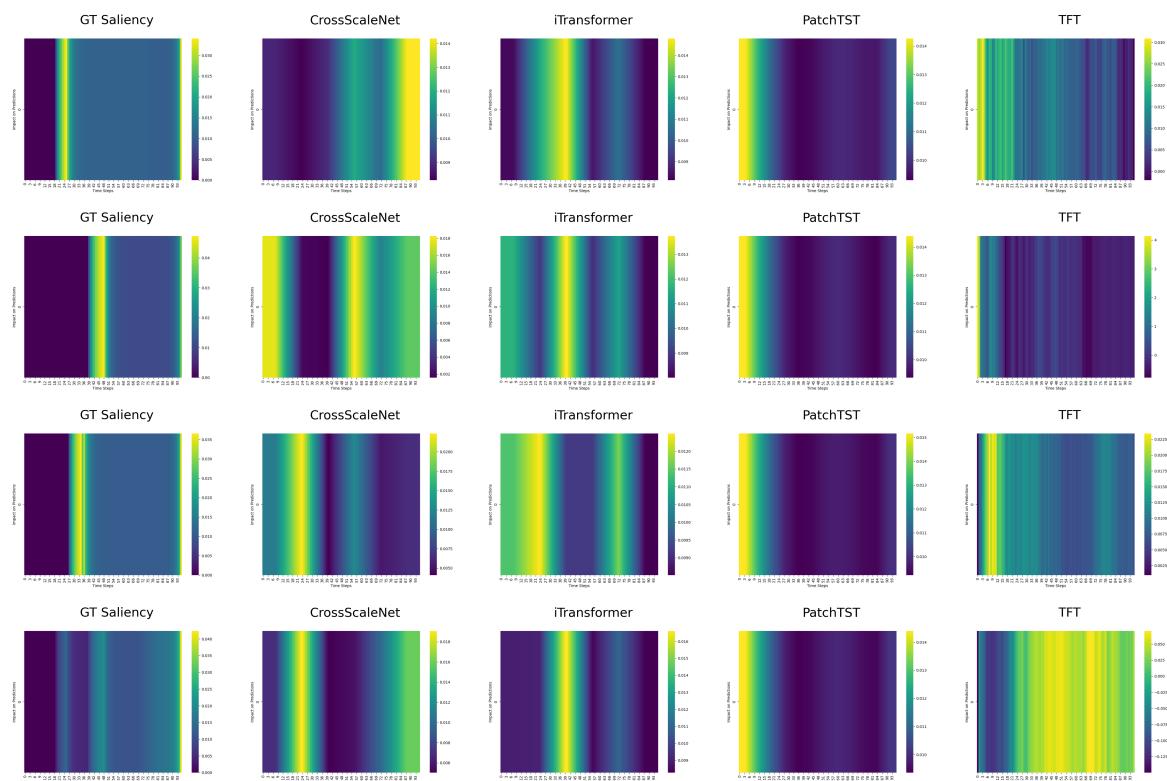


Figure 9: Visualization of ground truth and predicted saliency maps via different architectures for synthetic datasets: SYN5-6-7-8

References

- [1] Andreas Theissler, Francesco Spinnato, Udo Schlegel, and Riccardo Guidotti. Explainable ai for time series classification: a review, taxonomy and research directions. *Ieee Access*, 10:100700–100724, 2022.
- [2] Thomas Rojat, Raphaël Puget, David Filliat, Javier Del Ser, Rodolphe Gelin, and Natalia Díaz-Rodríguez. Explainable artificial intelligence (xai) on timeseries data: A survey. *arXiv preprint arXiv:2104.00950*, 2021.
- [3] Ilija Šimić, Vedran Sabol, and Eduardo Veas. Xai methods for neural time series classification: A brief review. *arXiv preprint arXiv:2108.08009*, 2021.
- [4] Scott Lundberg. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*, 2017.
- [5] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.
- [6] BN Oreshkin, D Carpov, N Chapados, and Y Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arxiv* 2019. *arXiv preprint arXiv:1905.10437*, 2019.
- [7] Jingyuan Wang, Ze Wang, Jianfeng Li, and Junjie Wu. Multilevel wavelet decomposition network for interpretable time series analysis. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2437–2446, 2018.
- [8] Bryan Lim, Sercan Ö Arik, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.
- [9] Yukai Ding, Yuelong Zhu, Jun Feng, Pengcheng Zhang, and Zirun Cheng. Interpretable spatio-temporal attention lstm model for flood forecasting. *Neurocomputing*, 403:348–359, 2020.
- [10] Sofia Serrano and Noah A Smith. Is attention interpretable? *arXiv preprint arXiv:1906.03731*, 2019.
- [11] Sarthak Jain and Byron C Wallace. Attention is not explanation. *arXiv preprint arXiv:1902.10186*, 2019.
- [12] Cristian Challu, Kin G Olivares, Boris N Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. Nhits: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 6989–6997, 2023.
- [13] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.
- [14] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024.
- [15] Ibrahim Delibasoglu, Sanjay Chakraborty, and Fredrik Heintz. Lms-autotsf: Learnable multi-scale decomposition and integrated autocorrelation for time series forecasting. *arXiv preprint arXiv:2412.06866*, 2024.
- [16] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.
- [17] G Woo, C Liu, D Sahoo, A Kumar, and S Hoi. Etsformer: Exponential smoothing transformer for time-series forecasting. *arxiv* 2022. *arXiv preprint arXiv:2202.01381*.
- [18] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.
- [19] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017.
- [20] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.
- [21] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arxiv*. *arXiv preprint arXiv:1706.03825*, 12, 2017.
- [22] Sangdi Lin and George C Runger. Gcrnn: Group-constrained convolutional recurrent neural network. *IEEE transactions on neural networks and learning systems*, 29(10):4709–4718, 2017.
- [23] Maël Guillemé, Véronique Masson, Laurence Rozé, and Alexandre Termier. Agnostic local explanation for time series classification. In *2019 IEEE 31st international conference on tools with artificial intelligence (ICTAI)*, pages 432–439. IEEE, 2019.
- [24] En-Yu Hsu, Chien-Liang Liu, and Vincent S Tseng. Multivariate time series early classification with interpretability using deep learning and attention mechanism. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 541–553. Springer, 2019.
- [25] Tsung-Yu Hsieh, Suhang Wang, Yiwei Sun, and Vasant Honavar. Explainable multivariate time series classification: a deep neural network which learns to attend to important variables as well as time intervals. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 607–615, 2021.
- [26] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.
- [27] Siyan Liu, Dan Lu, Daniel Ricciuto, and Anthony Walker. Improving net ecosystem co 2 flux prediction using memory-based interpretable machine learning. In *2022 IEEE international conference on data mining workshops (ICDMW)*, pages 1111–1119. IEEE, 2022.
- [28] Tian Guo, Tao Lin, and Nino Antulov-Fantulin. Exploring interpretable lstm neural networks over multi-variable data. In *International conference on machine learning*, pages 2494–2504. PMLR, 2019.
- [29] Dawei Zhou, Lecheng Zheng, Yada Zhu, Jianbo Li, and Jingrui He. Domain adaptive multi-modality neural attention network for financial forecasting. In *Proceedings of The Web Conference 2020*, pages 2230–2240, 2020.
- [30] Hugues Turbé, Mina Bjelogrlic, Christian Lovis, and Gianmarco Mengaldo. Evaluation of post-hoc interpretability methods in time-series classification. *Nature Machine Intelligence*, 5(3):250–260, 2023.

- [31] Yuyi Zhang, Qiushi Sun, Dongfang Qi, Jing Liu, Ruimin Ma, and Ovanes Petrosian. Shaptime: A general xai approach for explainable time series forecasting. In *Proceedings of SAI Intelligent Systems Conference*, pages 659–673. Springer, 2023.
- [32] Zhendong Wang, Ioanna Miliou, Isak Samsten, and Panagiotis Papapetrou. Counterfactual explanations for time series forecasting. In *2023 IEEE International Conference on Data Mining (ICDM)*, pages 1391–1396. IEEE, 2023.
- [33] JY Oostvogel. Interpretable deep learning for time series forecasting.
- [34] AR Troncoso-García, María Martínez-Ballesteros, Francisco Martínez-Álvarez, and Alicia Troncoso. A new approach based on association rules to add explainability to time series forecasting models. *Information Fusion*, 94:169–180, 2023.
- [35] Sanjay Chakraborty, Ibrahim Delibasoglu, and Fredrik Heintz. Edformer: Embedded decomposition transformer for interpretable multivariate time series predictions. *arXiv preprint arXiv:2412.12227*, 2024.
- [36] Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. Freeway performance measurement system: mining loop detector data. *Transportation research record*, 1748(1):96–102, 2001.
- [37] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.
- [38] Song Chen. Beijing PM2.5. UCI Machine Learning Repository, 2015. DOI: <https://doi.org/10.24432/C5JS49>.