Transparent Networks for Multivariate Time Series

Anonymous Author(s)
Affiliation
Address
email

Abstract

Transparent models, which are machine learning models that produce inherently interpretable predictions, are receiving significant attention in high-stakes domains. However, despite much real-world data being collected as time series, there is a lack of studies on transparent time series models. To address this gap, we propose a novel transparent neural network model for time series called Generalized Additive Time Series Model (GATSM). GATSM consists of two parts: 1) independent feature networks to learn feature representations, and 2) a transparent temporal module to learn temporal patterns across different time steps using the feature representations. This structure allows GATSM to effectively capture temporal patterns and handle dynamic-length time series while preserving transparency. Empirical experiments show that GATSM significantly outperforms existing generalized additive models and achieves comparable performance to black-box time series models, such as recurrent neural networks and Transformer. In addition, we demonstrate that GATSM finds interesting patterns in time series. The source code is available at `https://anonymous.4open.science/r/GATSM-78F4/`.

1 Introduction

Artificial neural networks excel at learning complex representations and demonstrate remarkable predictive performance across various fields. However, their complexity makes interpreting the decision-making processes of neural network models challenging. Consequently, post-hoc explainable artificial intelligence (XAI) methods, which explain the predictions of trained black-box models, have been widely studied in recent years [1, 2, 3, 4]. XAI methods are generally effective at providing humans with understandable explanations of model predictions. However, they may produce incorrect and unfaithful explanations of the underlying black-box model and cannot provide actual contributions of input features to model predictions [5, 6]. Therefore, their applicability to high-stakes domains-such as healthcare and fraud detection, where faithfulness to the underlying model and actual contributions of features are important-is limited.

Due to these limitations, transparent (i.e., inherently interpretable) models are attracting attention as alternatives to XAI in high-stakes domains [7, 8, 9]. Modern transparent models typically adhere to the *generalized additive model* (GAM) framework [10]. A GAM consists of independent functions, each corresponding to an input feature, and makes predictions as a linear combination of these functions (e.g., the sum of all functions). Therefore, each function reflects the contribution of its respective feature. For this reason, interpreting GAMs is straightforward, making them widely used in various fields, such as healthcare [11, 12], survival analysis [13], and model bias discovery [7, 14, 15]. However, despite much real-world data being collected as time series, research on GAMs for time series remains scarce. Consequently, the applicability of GAMs in real-world scenarios is still limited.

To overcome this limitation, we propose a novel transparent model for multivariate time series called Generalized Additive Time Series Model (GATSM). GATSM consists of independent feature networks to learn feature representations and a transparent temporal module to learn temporal patterns.

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

Since employing distinct networks across different time steps requires a massive amount of learnable parameters, the feature networks in GATSM share the weights across all time steps, while the temporal module independently learns temporal patterns. GATSM then generates final predictions by integrating the feature representations with the temporal information from the temporal module. This strategy allows GATSM to effectively capture temporal patterns and handle dynamic-length time series while preserving transparency. Additionally, this approach facilitates the separate extraction of time-independent feature contributions, the importance of individual time steps, and time-dependent feature contributions through the feature functions, temporal module, and final prediction. To demonstrate the effectiveness of GATSM, we conducted empirical experiments on various time series datasets. The experimental results show that GATSM significantly outperforms existing GAMs and achieves comparable performances to black-box time series models, such as recurrent neural networks and Transformer [16]. In addition, we provide visualizations of GATSM's predictions to demonstrate that GATSM finds interesting patterns in time series.

2 Related Works

Various XAI studies have been conducted over the past decade [7, 8, 9, 17, 18]; however, they are less relevant to the transparent model that is the subject of this study. Therefore, we refer readers to [19, 20] for more detailed information on recent XAI research. In this section, we review existing transparent models closely related to our GATSM and discuss their limitations.

Table 1: Advantages of GATSM.

	Time series input	Temporal pattern	Dynamic time series
existing GAMs			
NATM	✓		
GATSM (our)	✓	✓	✓

The simple linear model is designed to fit the conditional expectation $g\left(\mathbb{E}\left(y \mid \mathbf{x}\right)\right) = \sum_{i=1}^{M} x_i w_i$, where $g(\cdot)$ is a link function, M indicates the number of input features, y is the target value for the given input features $\mathbf{x} \in \mathbb{R}^M$, and $w_i \in \mathbb{R}$ is the learnable weight for x_i. This model captures only linear relationships between the target y and the inputs $\mathbf{x}$. To address this limitation, GAM [10] extends the simple linear model to the generalized form as follows:

$$g\left(\mathbb{E}\left(y \mid \mathbf{x}\right)\right) = \sum_{i=1}^{M} f_i\left(x_i\right), \tag{1}$$

where each $f_i(\cdot)$ is a function that models the effect of a single feature, referred as a feature function. Typically, $f_i\left(\cdot\right)$ becomes a non-linear function such as a decision tree or neural network to capture non-linear relationships.

Originally, GAMs were fitted via the backfitting algorithm using smooth splines [10, 21]. Later, Yin Lou et al. [22] and Harsha Nori et al. [23] have proposed boosted decision tree-based GAMs, which use boosted decision trees as feature functions. Spline- and tree-based GAMs have less flexibility and scalability. Thus, extending them to transfer or multi-task learning is challenging. To overcome this problem, various neural network-based GAMs have been proposed in recent years. Potts [24] introduced generalized additive neural network, which employs 2-layer neural networks as feature functions. Similarly, Rishabh Agarwal et al. [7] proposed neural additive model (NAM) that employs multi-layer neural networks. To improve the scalability of NAM, Chun-Hao Chang et al. [8] and Filip Radenovic et al. [9] proposed the neural oblivious tree-based GAM and the basis network-based GAM, respectively. Xu et al. [25] introduced a sparse version of NAM using the group LASSO. One disadvantage of GAMs is their limited predictive power, which stems from the fact that they only learn first-order feature interactions-i.e., relationships between the target value and individual features. To address this, various studies have been conducted to enhance the predictive powers of GAMs by incorporating higher-order feature interactions, while still maintaining transparency. GA^{2}M [26] simply takes pairwise features as inputs to learn pairwise interactions. GAMI-Net [27], a neural network-based GAM, consists of networks for main effects (i.e., first-order interactions) and pairwise interactions. To enhance the interpretability of GAMI-Net, the sparsity and heredity constraints are added, and trivial features are pruned in the training process. Sparse interaction additive network [28]

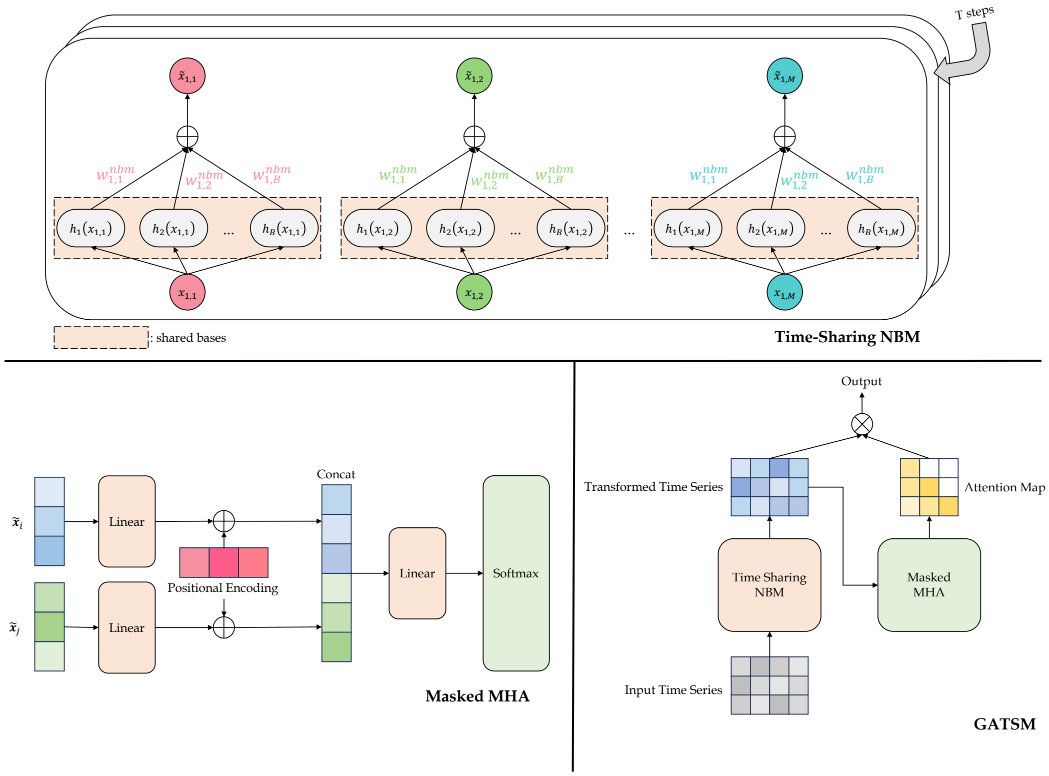


Figure 1: Architecture of GATSM.

is a 3-phase method for exploiting higher-order interactions. Initially, a black-box neural network is trained; subsequently, the top-k important features are identified using explainable feature attribution methods like LIME [1] and SHAP [2], and finally, NAM is trained with these extracted features. Dubey et al. [29] introduced scalable polynomial additive model, an end-to-end model that learns higher-order interactions via polynomials. Similarly, Kim et al. [15] proposed higher-order NAM that utilizes the feature crossing technique to capture higher-order interactions. Despite their capabilities, the aforementioned GAMs cannot process time series data, which limits their applicability in real-world scenarios. Recently, neural additive time series Model (NATM) [30], a time-series adaptation of NAM, has been proposed. However, NATM handles each time step independently with separate feature networks. This approach cannot capture effective temporal patterns and only takes fixed-length time series as input. Our GATSM not only captures temporal patterns but also handles dynamic-length time series. Table 1 shows the advantages of our GATSM compared to existing GAMs.

3 Problem Statement

We tackle the problem of the existing GAMs on time series. Equation (1) outlines the GAM framework for tabular data, which fails to capture the interactions between current and previous observations in time series. A straightforward method to extend GAM to time series, adopted in NATM, is applying distinct feature functions to each time step and summing them to produce predictions:

$$g\left(\mathbb{E}\left(y_t \mid \mathbf{X}_{:t}\right)\right) = \sum_{i=1}^{t} \sum_{j=1}^{M} f_{i,j}\left(x_{i,j}\right), \tag{2}$$

where $\mathbf{X} \in \mathbb{R}^{T \times M}$ is a time series with T time steps and M features, and t is the current time step. This method can handle time series data as input but fails to capture effective temporal patterns because the function $f_{i,j}\left(\cdot\right)$ still does not interact with previous time steps. To overcome this problem, we suggest a new form of GAM for time series defined as follows:

$$g\left(\mathbb{E}\left(y_t \mid \mathbf{X}_{:t}\right)\right)=\sum_{i=1}^{t}\sum_{j=1}^{M} f_{i,j}\left(x_{i,j}, \mathbf{X}_{:t}\right). \tag{3}$$

Definition 3.1 *GAMs for time series, which capture temporal patterns hold the form of Equation 3.*

In Equation (3), the function $f(\cdot,\cdot)$ can capture interactions between current and previous time steps. Therefore, GAMs adhering to Definition 3.1 are capable of capturing temporal patterns. However, implementing such a model while maintaining transparency poses challenges. In the following section, we will describe our approach to implementing a GAM that holds Definition 3.1. To the best of our knowledge, no existing literature addresses Definition 3.1.

4 Our Method: Generalized Additive Time Series Model

4.1 Architecture

Figure 1 shows the overall architecture of GATSM. Our model has two modules: 1) feature networks, called time-sharing neural basis model, for learning feature representations, and 2) masked multi-head attention for learning temporal patterns.

Time-Sharing NBM: Assume a time series with T time steps and M features. Applying GAMs to this time series necessitates $T \times M$ feature functions, which becomes problematic when dealing with large T or M due to increased model size. This limits the applicability of GAMs to real-world datasets. To overcome this problem, we extend neural basis model (NBM) [9] to time series as:

$$\tilde{x}_{i,j}=f_j\left(x_{i,j}\right)=\sum_{k=1}^{B} h_k\left(x_{i,j}\right) w_{j,k}^{nbm}. \tag{4}$$

We refer to this extended version of NBM as time-sharing NBM. Time-sharing NBM has B basis functions, with each basis $h_k(\cdot)$ taking a feature $x_{i,j}$ as input. The feature-specific weight $w_{j,k}^{nbm}$ then projects the basis to the transformed feature $\tilde{x}_{i,j}$. As depicted in Equation 4, the basis functions are shared across all features and time steps, drastically reducing the number of required feature functions $T \times M$ to B. We use $B=100$ and implement $h_k(\cdot)$ using multi-layer perceptron (MLP).

Masked MHA: GATSM employs multi-head attention (MHA) to learn temporal patterns. Although the dot product attention [16] is popular, simple dot operation has low expressive power [31]. Therefore, we adopt the 2-layer attention mechanism proposed by [31] to GATSM. We first transform $\tilde{\mathbf{x}}_i=[\tilde{x}_{i,1},\tilde{x}_{i,2},\cdots,\tilde{x}_{i,M}] \in \mathbb{R}^M$ produced by Equation 4 as follows:

$$\mathbf{v}_i=\tilde{\mathbf{x}}_i^{\intercal}\mathbf{Z}+\mathbf{pe}_i, \tag{5}$$

where $\mathbf{Z} \in \mathbb{R}^{M\times D}$ is a learnable weight, $\mathbf{pe}_i=[pe_{i,1},pe_{i,2},\cdots,pe_{i,D}] \in \mathbb{R}^D$ is the positional encoding for i-th step, and D indicates the hidden size. The positional encoding is defined as follows:

$$pe_{i,j}=\begin{cases}\sin\left(\frac{i}{10000^{2j/D}}\right) & \text{if } j \bmod 2=1,\\ \cos\left(\frac{i}{10000^{2j/D}}\right) & \text{otherwise.}\end{cases} \tag{6}$$

The positional encoding helps GATSM effectively capture temporal patterns. While learnable position embedding also works in GATSM, we recommend positional encoding because position embedding requires knowledge of the maximum number of time steps, which is often unknown in real-world settings. After computing $\mathbf{v}_i$, we calculate the attention scores as follows:

$$e_{k,i,j}=\sigma\left(\left[\mathbf{v}_i \mid \mathbf{v}_j\right]^{\intercal}\mathbf{w}_k^{attn}\right)m_{i,j}, \tag{7}$$

$$a_{k,i,j}=\frac{\exp\left(e_{k,i,j}\right)}{\sum_{t=1}^{T}\exp\left(e_{k,i,t}\right)}, \tag{8}$$

where k is attention head index, $\sigma(\cdot)$ is an activation function, $\mathbf{w}_k^{attn} \in \mathbb{R}^{2D}$, and $m_{i,j} \in \mathbb{R}$ is the mask value used to block future information. The time mask is defined as follows:

$$m_{i,j}=\begin{cases}1 & \text{if } i \leq j,\\ -\infty & \text{otherwise.}\end{cases} \tag{9}$$

Inference: The prediction of GATSM is produced by combining the transformed features from time-sharing NBM with the attention scores from masked MHA.

$$\hat{y}_t = \sum_{k=1}^{K} \mathbf{a}_{k,t}^{\intercal} \tilde{\mathbf{X}} \mathbf{w}_k^{out}, \tag{10}$$

where K is the number of attention heads, $\mathbf{a}_{k,t} = [a_{k,i,1}, a_{k,i,2}, \cdots, a_{k,i,T}] \in \mathbb{R}^T$ is the attention map in Equation 8, $\tilde{\mathbf{X}} = [\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \cdots, \tilde{\mathbf{x}}_T] \in \mathbb{R}^{T \times M}$ is the transformed features in Equation 4, and $\mathbf{w}_k^{out} \in \mathbb{R}^M$ is the learnable output weight.

Interpretability: We can rewrite Equation 10 as the following scalar form:

$$\begin{aligned} \sum_{k=1}^{K} \mathbf{a}_{k,t}^{\intercal} \tilde{\mathbf{X}} \mathbf{w}_k^{out} &= \sum_{u=1}^{t} \sum_{m=1}^{M} \sum_{k=1}^{K} \sum_{b=1}^{B} a_{k,t,u} h_b \left(x_{t,m} \right) w_{m,b}^{nbm} w_{k,m}^{out} \\ &= \sum_{u=1}^{t} \sum_{m=1}^{M} f_{u,m} \left(x_{u,m}, \mathbf{X}_{:t} \right) \end{aligned} \tag{11}$$

Equation 11 shows that GATSM satisfying Definition 3.1. We can derive three types of interpretations from GATSM: 1) $a_{k,t,u}$ indicates the importance of time step u at time step t, 2) $h_b \left(x_{t,m} \right) w_{m,b}^{nbm} w_{k,m}^{out}$ represents the time-independent contribution of feature m, and 3) $a_{k,t,u} h_b \left(x_{t,m} \right) w_{m,b}^{nbm} w_{k,m}^{out}$ represents the time-dependent contribution of feature m at time step t.

5 Experiments

5.1 Experimental Setup

Datasets: We conducted our experiments using eight publicly available real-world time series datasets. From the Monash repository [32], we sourced three datasets: Energy, Rainfall, and AirQuality. Another three datasets, Heartbeat, LSST, and NATOPS, were downloaded from the UCR repository [33]. The remaining two datasets, Mortality and Sepsis, were downloaded from the PhysioNet [34]. We perform ordinal encoding for categorical features and standardize features to have zero-mean and unit-variance. For forecasting tasks, target value y is also standardized to zero-mean and unit-variance. If the dataset contains missing values, we impute categorical features with their modes and numerical features with their means. The dataset is split into a 60%/20%/20% ratio for training, validation, and testing, respectively. Table 2 shows the statistics of the experimental datasets. Further details of the experimental datasets can be found in Appendix B.

Table 2: Dataset statistics.

Dataset	Task	Variable length	# of time series	Avg. length	# of features	# of classes
Energy	1-step FCST	No	137	24	24	-
Rainfall	1-step FCST	No	160,267	24	3	-
AirQuality	1-step FCST	No	16,966	24	9	-
Heartbeat	Binary	No	409	405	61	2
Mortality	Binary	Yes	12,000	49.861	41	2
Sepsis	Binary	Yes	40,336	38.482	40	2
LSST	Multi-class	No	4,925	36	6	14
NATOPS	Multi-class	No	360	51	24	6

FCST: forecasting

Baselines: We compare our GATSM with 12 baselines, which can be categorized into four groups: 1) Black-box tabular models include extreme gradient boosting (XGBoost) [35] and MLP. 2) Black-box time series models include simple recurrent neural network (RNN), gated recurrent unit (GRU), long short-term memory (LSTM), and Transformer [16]. 3) Transparent tabular models are simple linear model (Linear), explainable boosting machine (EBM) [23], NAM [7], NodeGAM [8], and NBM [9]. 4) NATM [30] is a transparent time series model.

Implementation: We implement XGBoost and EBM models using the `xgboost` and `interpretml` libraries, respectively. For NodeGAM, we employ the official implementation provided by its authors [8]. The remaining models are developed using PyTorch [36]. All models undergo hyperparameter

Table 3: Predictive performance comparison of various models.

Model Type	Model	Energy	Rainfall	AirQuality	Heartbeat	Mortality	Sepsis	LSST	NATOPS	Avg. Rank
Black-box Tabular Model	XGBoost	0.094 (±0.137)	0.002 (±0.002)	0.532 (±0.019)	0.679 (±0.094)	0.707 (±0.015)	**0.816** (±0.007)	0.424 (±0.012)	0.200 (±0.049)	8.500 (±4.000)
	MLP	0.459 (±0.101)	0.011 (±0.004)	0.423 (±0.031)	0.654 (±0.082)	0.842 (±0.014)	0.786 (±0.007)	0.417 (±0.008)	0.211 (±0.065)	7.375 (±2.134)
Black-box Time Series Model	RNN	0.320 (±0.122)	0.068 (±0.020)	0.644 (±0.032)	0.661 (±0.078)	0.581 (±0.040)	0.782 (±0.009)	0.422 (±0.029)	0.592 (±0.110)	7.750 (±2.712)
	GRU	0.435 (±0.107)	0.089 (±0.034)	0.701 (±0.018)	0.694 (±0.052)	0.818 (±0.014)	0.785 (±0.010)	0.629 (±0.013)	0.931 (±0.045)	4.375 (±2.669)
	LSTM	0.359 (±0.112)	0.090 (±0.031)	0.683 (±0.026)	0.648 (±0.042)	0.790 (±0.020)	0.779 (±0.008)	0.491 (±0.082)	0.908 (±0.035)	6.375 (±3.623)
	Transformer	0.263 (±0.263)	**0.098** (±0.035)	**0.711** (±0.027)	0.690 (±0.040)	0.844 (±0.019)	0.789 (±0.010)	**0.679** (±0.019)	**0.967** (±0.029)	4.000 (±3.703)
Transparent Tabular Model	Linear	0.482 (±0.112)	0.004 (±0.001)	0.241 (±0.019)	0.637 (±0.070)	0.838 (±0.017)	0.723 (±0.011)	0.311 (±0.010)	0.206 (±0.045)	10.125 (±3.871)
	EBM	-0.200 (±0.409)	0.004 (±0.001)	0.324 (±0.014)	0.666 (±0.056)	0.729 (±0.017)	0.802 (±0.011)	0.408 (±0.016)	0.164 (±0.053)	9.750 (±3.284)
	NAM	0.363 (±0.218)	0.006 (±0.002)	0.300 (±0.013)	0.645 (±0.026)	0.853 (±0.014)	0.800 (±0.006)	0.400 (±0.011)	0.242 (±0.040)	7.875 (±3.643)
	NodeGAM	0.398 (±0.195)	0.006 (±0.002)	0.380 (±0.032)	0.681 (±0.046)	**0.854** (±0.013)	0.802 (±0.007)	0.400 (±0.028)	0.247 (±0.012)	6.375 (±3.623)
	NBM	0.330 (±0.251)	0.007 (±0.003)	0.301 (±0.012)	0.716 (±0.039)	0.852 (±0.014)	0.799 (±0.006)	0.388 (±0.014)	0.189 (±0.029)	7.875 (±3.603)
Transparent Time Series Model	NATM	0.304 (±0.122)	0.038 (±0.011)	0.548 (±0.028)	0.724 (±0.043)	N/A	N/A	0.452 (±0.010)	0.878 (±0.058)	5.667 (±2.582)
	GATSM (ours)	**0.493** (±0.173)	0.073 (±0.027)	0.583 (±0.026)	**0.843** (±0.025)	0.853 (±0.015)	0.797 (±0.007)	0.570 (±0.024)	0.956 (±0.027)	**3.125** (±1.808)

tuning via Optuna [37]. The pytorch-based models are optimized with the Adam with decoupled weight decay (AdamW) [38] optimizer on an NVIDIA A100 GPU. Model training is halted if the validation loss does not decrease over 20 epochs. We use mean squared error for the forecasting tasks, and for classification tasks, we use cross-entropy loss. Further details of the model implementations and hyper-parameters are provided in Appendix C.

5.2 Comparison with baselines

Table 3 shows the predictive performances of the experimental models. We report mean scores and standard deviations over five different random seeds. For the forecasting datasets, we evaluate R^2 scores. For the binary classification datasets, we assess the area under the receiver operating characteristic curve (AUROC). For the multi-class classification datasets, we measure accuracy. We highlight the best-performing model in **bold** and underline the second-best model. Since the tabular models cannot handle time series, they only take $\mathbf{x}_t$ to produce y_t.

On the Energy and Heartbeat datasets, which are small in size, our GATSM demonstrates the best performance, indicating strong generalization ability. EBM, XGBoost, and Transformer struggle with overfitting on the Energy dataset. For the Mortality and Sepsis datasets, there is no significant performance difference between tabular and time series models, nor between black-box and transparent models. This suggests that these two healthcare datasets lack significant temporal patterns and feature interactions. It is likely that seasonal patterns are hard to detect in medical data, and the patient's current condition already encapsulates previous conditions, making historical data less crucial. Since these datasets contain variable-length time series, the performance of NATM, which can only handle fixed-length time series, is not available. On the Rainfall, AirQuality, LSST, and NATOPS datasets, the time series models significantly outperform the tabular models, indicating that these datasets contain important temporal patterns that tabular models cannot capture. Additionally, the black-box models outperform the transparent models, suggesting that these datasets have higher-order feature interactions that transparent models cannot capture. Nevertheless, GATSM is the best model within the transparent model group and performs comparably to Transformer. Overall, GATSM achieved the best average rank in the experiments, followed by the Transformer, indicating GATSM's superiority. Additional experiments on model throughput and an ablation study on the basis functions are presented in Appendix D.

Table 4: Ablation study on different feature functions.

Feature Function	Energy	Rainfall	AirQuality	Heartbeat	Mortality	Sepsis	LSST	NATOPS
Linear	0.283($\pm$0.277)	0.071($\pm$0.024)	0.563($\pm$0.019)	0.766($\pm$0.024)	0.832($\pm$0.015)	0.735($\pm$0.012)	0.398($\pm$0.030)	**0.972**($\pm$0.020)
NAM	0.304($\pm$0.229)	0.068($\pm$0.021)	0.564($\pm$0.019)	0.838($\pm$0.032)	0.851($\pm$0.013)	**0.801**($\pm$0.005)	0.553($\pm$0.023)	0.933($\pm$0.039)
NBM	**0.493**($\pm$0.173)	**0.073**($\pm$0.027)	**0.583**($\pm$0.026)	**0.843**($\pm$0.025)	**0.853**($\pm$0.015)	0.797($\pm$0.007)	**0.570**($\pm$0.024)	0.956($\pm$0.027)

Table 5: Ablation study on the temporal module.

Temporal Module	Energy	Rainfall	AirQuality	Heartbeat	Mortality	Sepsis	LSST	NATOPS
Base	0.452($\pm$0.087)	0.007($\pm$0.002)	0.299($\pm$0.012)	0.661($\pm$0.043)	**0.854**($\pm$0.013)	0.798($\pm$0.008)	0.392($\pm$0.006)	0.192($\pm$0.027)
Base + PE	0.397($\pm$0.054)	0.007($\pm$0.003)	0.299($\pm$0.012)	0.681($\pm$0.068)	0.852($\pm$0.013)	**0.799**($\pm$0.007)	0.385($\pm$0.027)	0.228($\pm$0.029)
Base + MHA	0.368($\pm$0.230)	0.048($\pm$0.017)	0.555($\pm$0.020)	0.821($\pm$0.044)	0.847($\pm$0.020)	0.779($\pm$0.033)	**0.595**($\pm$0.013)	0.856($\pm$0.059)
Base + PE + MHA	**0.493**($\pm$0.173)	**0.073**($\pm$0.027)	**0.583**($\pm$0.026)	**0.843**($\pm$0.025)	0.853($\pm$0.015)	0.797($\pm$0.007)	0.570($\pm$0.024)	**0.956**($\pm$0.027)

5.3 Ablation study

Choice of feature function: We evaluate the performance of GATSM by changing the feature functions using three models: Linear, NAM, and NBM. Table 4 presents the results of this experiment. The simple linear function performs poorly because it lacks the capability to capture non-linear relationships. In contrast, NAM, which can capture non-linearity, shows improved performance over the linear function. However, NBM stands out by achieving the best performance in six out of eight datasets. This indicates that the basis strategy of NBM is highly effective for time series data.

Design of temporal module: We evaluate the performance of GATSM by modifying the design of the temporal module. The results are presented in Table 5. GATSM without the temporal module (Base) fails to learn temporal patterns and shows poor performance in the experiment. GATSM with only positional encoding (Base + PE) also shows similar performance to the Base, indicating that positional encoding alone is insufficient for capturing effective temporal patterns. GATSM with only multi-head attention (Base + MHA) outperforms the previous two methods, demonstrating that the MHA mechanism is beneficial for capturing temporal patterns. Finally, our full GATSM (Base + PE + MHA) significantly outperforms the other methods, suggesting that the combination of PE and MHA creates a synergistic effect. Consistent with our previous findings in section 5.2, all four methods show similar performances on the Mortality and Sepsis datasets, which lack significant temporal patterns.

5.4 Interpretation

In this section, we visualize four interpretations of GATSM's predictions on the AirQuality dataset. In addition, interpretations for the Rainfall dataset can be found in Appendix E.

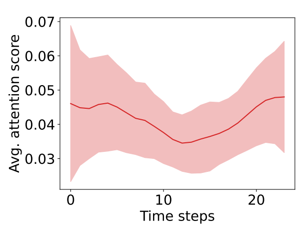


Figure 2: Average attention scores of time steps on the AirQuality dataset.

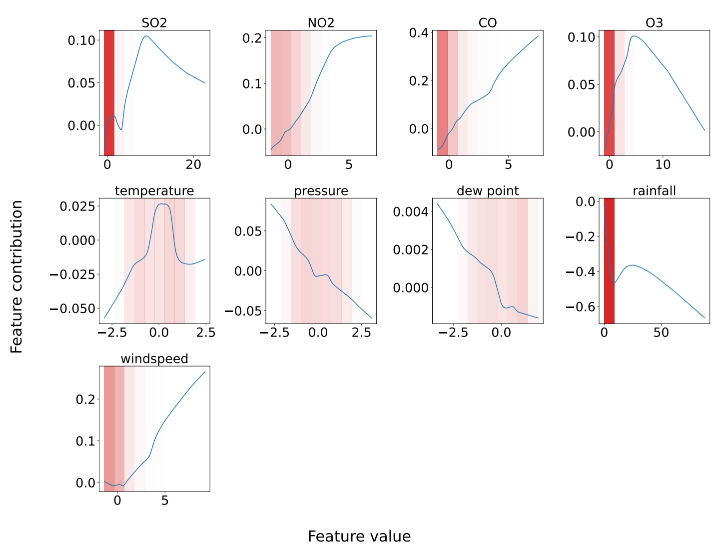


Figure 3: Global interpretations of features in the Air Quality dataset.

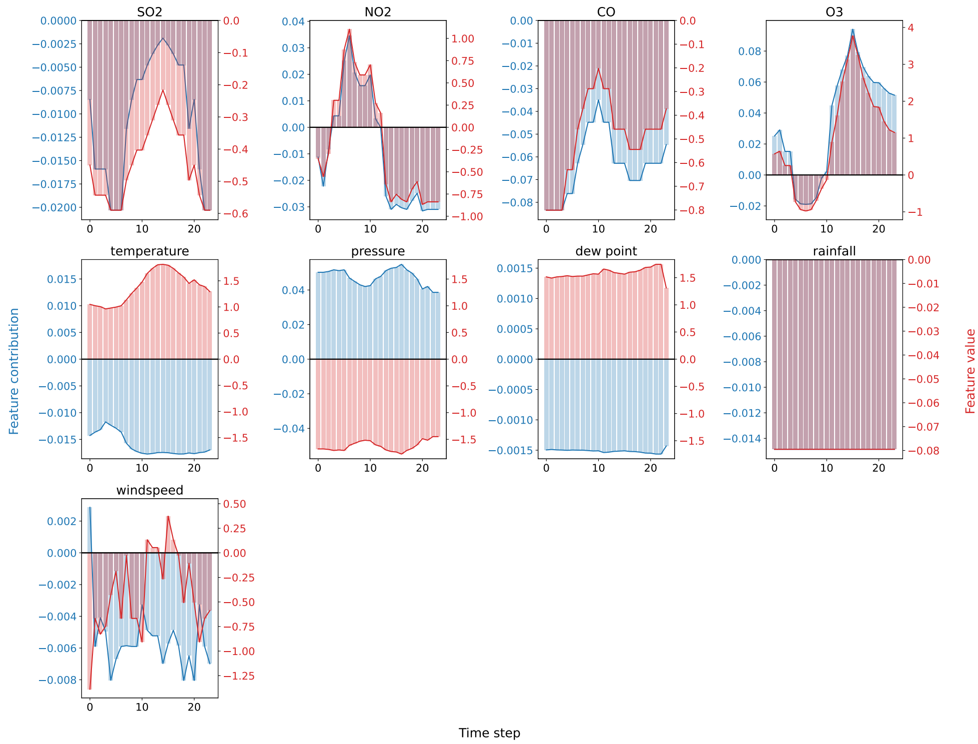


Figure 4: Local time-independent feature contributions.

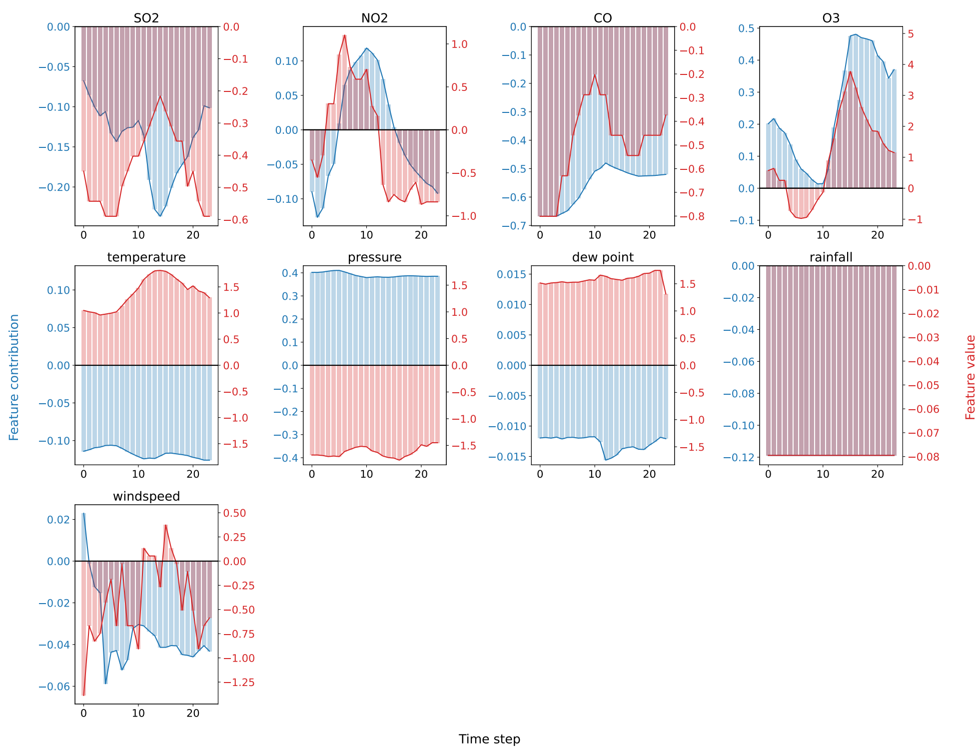


Figure 5: Local time-dependent feature contributions.

Time-step importance: We plot the average attention scores at the last time step T in Figure 2. The process for extracting the average attention score of time step u at time step t is formalized as $\sum_{k=1}^{K} a_{k,t,u}$. This process is repeated over all data samples, and the results are averaged. Based on Figure 2, it seems that GATSM pays more attention to the initial and last states than to the intermediate states. This indicates that the current concentration of particulate matter depends on the initial state.

Global feature contribution: Figure 3 illustrates the global behavior of features in the AirQuality dataset, with red bars indicating the density of training samples. We extract $\sum_{k=1}^{K} h_b(x_{t,m}) w_{m,b}^{nbm} w_{k,m}^{out}$ from GATSM and repeat this process over the range of minimum to maximum feature values to plot the line. We found that the behavior of *SO2*, *O3*, and *windspeed* is inconsistent with prior human knowledge. Typically, high levels of *SO2* and *O3* are associated with poor air quality. However, GATSM learned that particulate matter concentration starts to decrease when *SO2* exceeds 10 and *O3* exceeds 5. This discrepancy may be due to sparse training samples in these regions, leading to insufficient training, or there may be interactions with other features. Another known fact is that high *windspeed* decreases particulate matter concentration. This is consistent when *windspeed* is below 0.7 in our observation. However, particulate matter concentration drastically increases when *windspeed* exceeds 0.7, likely due to the wind causing yellow dust.

Local time-independent feature contribution: To interpret the prediction of a data sample, we plot the local time-independent feature contributions, $\sum_{k=1}^{K} h_b(x_{t,m}) w_{m,b}^{nbm} w_{k,m}^{out}$, in Figure 4. The main x-axis (blue) represents feature contribution, the sub x-axis (red) represents feature value, and the y-axis represents time steps. We found that *SO2*, *NO2*, *CO*, and *O3* have positive correlations. In contrast, *temperature*, *pressure*, *dew point*, and *windspeed* have negative correlations. These are consistent with the global interpretations shown in Figure 3. Rainfall has the same values across all time steps.

Local time-dependent feature contribution: We also visualize the local time-dependent feature contributions, $\sum_{k=1}^{K} a_{k,t,u} h_b(x_{t,m}) w_{m,b}^{nbm} w_{k,m}^{out}$. Figure 5 illustrates the interpretation of the same data sample as in Figure 4. The time-dependent interpretation differs slightly from the time-independent interpretation. We found that there are time lags in *SO2*, *NO2*, *CO*, and *O3*, meaning previous feature values affect current feature contributions. For example, in the case of *SO2*, low feature values around time step 5 lead to low feature contributions around time step 13.

6 Future Works & Conclusion

Although GATSM achieved state-of-the-art performance within the transparent model category, it has several limitations. This section discusses these limitations and suggests future work to address them. GAMs have relatively slower computational times and larger model sizes compared to black-box models because they require the same number of feature functions as input features. To address this problem, methods such as the basis strategy can be proposed to reduce the number of feature functions, or entirely new methods for transparent models can be developed. The attention mechanism in GATSM may be a bottleneck. Fast attention mechanisms proposed in the literature [39, 40, 41, 42, 43], or the recently proposed Mamba [44], can help overcome this limitation. Existing time series models, including GATSM, only handle discrete time series and have limited length generalization ability, resulting in significantly reduced performance when very long sequences, unseen during training, are input. Extending GATSM to continuous models using NeuralODE [45] or HiPPO [46] could address this issue. GATSM still cannot learn higher-order feature interactions internally and shows low performance on complex datasets. Feature interaction methods proposed for transparent models may help address this problem [29, 15].

In this papre, we proposed a novel transparent model for time series named GATSM. GATSM consists of time-sharing NBM and the temporal module to effectively learn feature representations and temporal patterns while maintaining transparency. The experimental results demonstrated that GATSM has superior generalization ability and is the only transparent model with performance comparable to Transformer. We provided various visual interpretations of GATSM, demonstrated that GATSM capture interesting patterns in time series data. We anticipate that GATSM will be widely adopted in various fields and demonstrate strong performance. The broader impacts of GATSM across various fields can be found in Appendix A.

References

[1] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[2] Scott M. Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, 2017.

[3] 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. 2017.

[4] Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 2020.

[5] Cynthia Rudin. Please Stop Explaining Black Box Models for High Stakes Decisions. In *Advances in Neural Information Processing Systems, Workshop on Critiquing and Correcting Trends in Machine Learning*, 2018.

[6] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1:206–215, May 2019.

[7] Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E. Hinton. Neural Additive Models: Interpretable Machine Learning with Neural Nets. In *Advances in Neural Information Processing Systems*, 2021.

[8] Chun-Hao Chang, Rich Caruana, and Anna Goldenberg. NODE-GAM: Neural Generalized Additive Model for Interpretable Deep Learning. In *International Conference on Learning Representations*, 2022.

[9] Filip Radenovic, Abhimanyu Dubey, and Dhruv Mahajan. Neural Basis Models for Interpretability. In *Advances in Neural Information Processing Systems*, 2022.

[10] Trevor Hastie and Robert Tibshirani. Generalized Additive Models. *Statistical Science*, 1(3): 297–318, August 1986.

[11] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noémie Elhadad. Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015.

[12] Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. How Interpretable and Trustworthy are GAMs? In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2021.

[13] Lev V. Utkin, Egor D. Satyukov, and Andrei V. Konstantinov. SurvNAM: The machine learning survival model explanation. *Neural Networks*, 147:81–102, March 2022.

[14] Sarah Tan, Rich Caruana, Giles Hooker, and Yin Lou. Distill-and-Compare: Auditing Black-Box Models Using Transparent Model Distillation. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 2018.

[15] Minkyu Kim, Hyun-Soo Choi, and Jinho Kim. Higher-order Neural Additive Models: An Interpretable Machine Learning Model with Feature Interactions. *arXiv preprint arXiv:2209.15409*, 2022.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, 2017.

[17] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLoS ONE*, 10(7), July 2015.

[18] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning Important Features Through Propagating Activation Differences. In *International Conference on Machine Learning*, 2017.

[19] Sajid Ali, Tamer Abuhmed, Shaker El-Sappagh, Khan Muhammad, Jose M. Alonso-Moral, Roberto Confalonieri, Riccardo Guidotti, Javier Del Ser, Natalia Díaz-Rodríguez, and Francisco

Herrera. Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence. *Information Fusion*, 99:101805, November 2023.

[20] Vikas Hassija, Vinay Chamola, Atmesh Mahapatra, Abhinandan Singal, Divyansh Goel, Kaizhu Huang, Simone Scardapane, Indro Spinelli, Mufti Mahmud, and Amir Hussain. Interpreting Black-Box Models: A Review on Explainable Artificial Intelligence. *Cognitive Computation*, 16(1):45–74, January 2024.

[21] Grace Wahba. *Spline Models for Observational Data*. SIAM, September 1990.

[22] Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible Models for Classification and Regression. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012.

[23] Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. InterpretML: A Unified Framework for Machine Learning Interpretability. *arXiv preprint arXiv:1909.09223*, 2019.

[24] William J. E. Potts. Generalized Additive Neural Networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1999.

[25] Shiyun Xu, Zhiqi Bu, Pratik Chaudhari, and Ian J. Barnett. Sparse Neural Additive Model: Interpretable Deep Learning with Feature Selection via Group Sparsity. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2023.

[26] Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. Accurate intelligible models with pairwise interactions. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013.

[27] Zebin Yang, Aijun Zhang, and Agus Sudjianto. GAMI-Net: An Explainable Neural Network based on Generalized Additive Models with Structured Interactions. *Pattern Recognition*, 120: 108192, December 2021.

[28] James Enouen and Yan Liu. Sparse Interaction Additive Networks via Feature Interaction Detection and Sparse Selection. *Advances in Neural Information Processing Systems*, 35, 2022.

[29] Abhimanyu Dubey, Filip Radenovic, and Dhruv Mahajan. Scalable Interpretability via Polynomials. *Advances in Neural Information Processing Systems*, 2022.

[30] Wonkeun Jo and Dongil Kim. Neural additive time-series models: Explainable deep learning for multivariate time-series prediction. *Expert Systems with Applications*, 228:120307, October 2023.

[31] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *International Conference on Learning Representations*, 2018.

[32] Chang Wei Tan, Christoph Bergmeir, Francois Petitjean, and Geoffrey I. Webb. Monash University, UEA, UCR Time Series Extrinsic Regression Archive. *arXiv preprint arXiv:2006.10996*, 2020.

[33] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The UEA multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

[34] Ary L. Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. PhysioBank, PhysioToolkit, and PhysioNet. *Circulation*, 101(23):e215–e220, June 2000.

[35] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[36] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, 2019.

[37] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.

[38] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019.

[39] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *International Conference on Machine Learning*, 2020.

[40] Lovish Madaan, Srinadh Bhojanapalli, Himanshu Jain, and Prateek Jain. Treeformer: Dense Gradient Trees for Efficient Attention Computation. In *International Conference on Learning Representations*, 2023.

[41] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-Attention with Linear Complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[42] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking Attention with Performers. In *International Conference on Learning Representations*, 2021.

[43] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The Efficient Transformer. In *International Conference on Learning Representations*, 2020.

[44] Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces, 2023.

[45] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, 2018.

[46] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. HiPPO: Recurrent Memory with Optimal Polynomial Projections. In *Advances in Neural Information Processing Systems*, 2020.

[47] Eric J. Pedersen, David L. Miller, Gavin L. Simpson, and Noam Ross. Hierarchical generalized additive models in ecology: an introduction with mgcv. *PeerJ*, 7:e6876, May 2019.

[48] Trevor Hastie and Robert Tibshirani. Generalized additive models for medical research. *Statistical Methods in Medical Research*, 4(3):187–196, September 1995.

[49] Appliances Energy Dataset, 2020. URL `https://doi.org/10.5281/zenodo.3902637`.

[50] Australia Rainfall Dataset, 2020. URL `https://doi.org/10.5281/zenodo.3902654`.

[51] Beijing PM10 Dataset, 2020. URL `https://doi.org/10.5281/zenodo.3902667`.

[52] Classification of Heart Sound Recordings: The PhysioNet/Computing in Cardiology Challenge 2016, 2016. URL `https://physionet.org/content/challenge-2016/1.0.0/`.

[53] Predicting Mortality of ICU Patients: The PhysioNet/Computing in Cardiology Challenge 2012, 2012. URL `https://physionet.org/content/challenge-2012/1.0.0/`.

[54] Early Prediction of Sepsis from Clinical Data: The PhysioNet/Computing in Cardiology Challenge 2019, 2019. URL `https://physionet.org/content/challenge-2019/1.0.0/`.

[55] PLAsTiCC Astronomical Classification, 2018. URL `https://www.kaggle.com/c/PLAsTiCC-2018`.

[56] AALTD'16 Time Series Classification Contest, 2016. URL `https://aaltd16.irisa.fr/challenge/`.

[57] Ignacio Oguiza. tsai - a state-of-the-art deep learning library for time series and sequential data. Github, 2023. URL `https://github.com/timeseriesAI/tsai`.

[58] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, 2012.

[59] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[60] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for Hyper-Parameter Optimization. In *Advances in Neural Information Processing Systems*, 2011.

[61] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. *arXiv preprint arXiv:2205.14135*, 2022.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of our work are described in section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: Our work does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provided experimental setup and implementation details in section 5.1 and Appendix C.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We used public datasets and opened our code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We described the experimental setting in section 5.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We provided standard deviations with experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provided information on the computational resource used in the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our work conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discussed the potential impacts of GATSM in Appendix A.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our work poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cited the used codes and data.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We opened the source code of GATSM, and the document to run the code is provided along with the code.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

A Broader impact

We discuss the expected impacts of GATSM across various fields.

- **Time series adaptation:** GATSM extends existing GAMs to time series, enabling tasks that traditional GAMs could not perform in this context - e.g., better performance on time series and finding temporal patterns.
- **Improved decision-making system:** GATSM can show users their exact decision-making process, providing trust and confidence in its predictions to users. This enables decision-makers to make more informed choices, crucial in high-stakes domains such as healthcare.
- **Ethical AI:** GATSM can examine that their outcomes are biased or discriminatory by displaying the shape of feature functions. This is particularly important in ethically sensitive domains, such as recidivism prediction.
- **Scientific discovery:** Transparent models have already been used in various research fields for scientific discovery [47, 48]. GATSM also can be applied to these domains to obtain novel scientific insights.

Despite these advantages, it is important to remember that the interpretations of transparent models do not necessarily reflect exact causal relationships. While transparent models provide clear and faithful interpretations, they are still not capable of identifying causal relationships. Causal discovery is a complex task that requires further research.

B Dataset details

We use eight publicly available datasets for our experiments. Three datasets - Energy, Rainfall, and AirQuality - can be downloaded from the Monash repository [32]. Another three datasets - Heartbeat, LSST, and NATOPS - are available from the UCR repository [33]. The remaining two datasets can be downloaded from the PhysioNet [34]. Details of the datasets are provided below:

- **Energy** [49]: This dataset consists of 24 features related to temperature and humidity from sensors and weather conditions. These features are measured every 10 minutes. The goal of this dataset is to predict total energy usage.
- **Rainfall** [50]: This dataset consists of temperatures measured hourly. The goal of this dataset is to predict total daily rainfall in Australia.
- **AirQuality** [51]: This dataset consists of features related to air pollutants and meteorological data. The goal of this dataset is to predict the PM10 level in Beijing.
- **Heartbeat** [52]: This dataset consists of heart sounds collected from various locations on the body. Each sound was truncated to five seconds, and a spectrogram of each instance was created with a window size of 0.061 seconds with a 70% overlap. The goal of this dataset is to classify the sounds as either normal or abnormal.
- **Mortality** [53] This dataset consists of records of adult patients admitted to the ICU. The input features include the patient demographics, vital signs, and lab results. The goal of this dataset is to predict the in-hospital death of patients.
- **Sepsis** [54]: This dataset consists of records of ICU patients. The input features include patient demographics, vital signs, and lab results. The goal of this dataset is to predict sepsis six hours in advance at every time step.
- **LSST** [55]: This challenge dataset aims to classify astronomical time series. These time series consist of six different light curves, simulated based on the data expected from the Large Synoptic Survey Telescope (LSST).
- **NATOPS** [56]: This dataset aims to classify the Naval Air Training and Operating Procedures Standardization (NATOPS) motions used to control aircraft movements. It consists of 24 features representing the x, y, and z coordinates for each of the eight sensor locations attached to the body.

We used `get_UCR_data()` and `get_Monash_regression_data()` functions in the `tsai` library [57] to load the UCR and Monash datasets.

Table 6: Optimal hyper-parameters for GATSM.

GATSM: [256, 256, 128] hidden dims, 100 basis functions								
Dataset	Batch Size	NBM Batch Norm.	NBM Dropout	Attn. Embedding Size	Attn. Heads	Attn. Dropout	Learning Rate	Weight Decay
Energy	32	False	2.315e-1	110	8	6.924e-2	4.950e-3	1.679e-3
Rainfall	32,768	False	5.936e-3	44	7	1.215e-3	9.225e-3	2.204e-6
AirQuality	4,096	False	2.340e-2	81	8	1.169e-1	6.076e-3	5.047e-6
Heartbeat	64	True	1.749e-1	92	2	1.653e-1	8.061e-3	4.787e-6
Mortality	512	False	7.151e-2	125	8	7.324e-1	7.304e-3	2.181e-4
Sepsis	512	True	6.523e-2	90	6	8.992e-1	4.509e-3	2.259e-2
LSST	1,024	False	2.500e-2	59	7	2.063e-1	5.561e-2	5.957e-3
NATOPS	64	True	4.827e-3	49	8	7.920e-1	8.156e-3	2.748e-2

C Implementation details

We use 13 models, including GATSM, for our experiments. We implement XGBoost and EBM using the `xgboost` [35] and `interpretml` [23] libraries, respectively. For NodeGAM, we employ the official implementation provided by its authors [8]. The remaining models are developed using PyTorch [36]. In addition, we implement the feature functions in NAM and NBM using grouped convolutions [58, 59] to enhance their efficiency. XGBoost and EBM are trained on two AMD EPYC 7513 CPUs, while the other models are trained on an NVIDIA A100 GPU with 80GB VRAM. All models undergo hyperparameter tuning via Optuna [37] with the Tree-structured Parzen Estimator (TPE) algorithm [60] in 100 trials. The hyperparameter search space and the optimal hyperparameters for the models are provided below:

- **XGBoost:** We tune the `n_estimators` in the integer interval [1, 1000], `max_depth` in the integer interval [0, 2000], learning rate in the continuous interval [1e-6, 1], `subsample` in the continuous interval [0, 1], and `colsample_bytree` in the continuous interval [0, 1].
- **MLP, NAM, NBM and NATM:** We tune the `batchnorm` in the descret set {False, True}, `dropout` in the continuous interval [0, 0.9], `learning_rate` in the continuous interval [1e-3, 1e-2], and `weight_decay` in the continuous interval [1e-6, 1e-1] on a log scale.
- **RNN, GRU and LSTM:** We tune the `hidden_size` in the integer interval [8, 128], `dropout` in the continuous interval [0, 0.9], `learning_rate` in the continuous interval [1e-3, 1e-2], and `weight_decay` in the continuous interval [1e-6, 1e-1] on a log scale.
- **Transformer:** We tune the `n_layers` in the integer interval [1, 4], `emb_size` in the integer interval [8, 32], `hidden_size` in the integer interval [8, 128], `n_heads` in the integer interval [1, 8], `dropout` in the continuous interval [0, 0.9], `learning_rate` in the continuous interval [1e-3, 1e-2], and `weight_decay` in the continuous interval [1e-6, 1e-1] on a log scale.
- **Linear:** We tune the `learning_rate` in the continuous interval [1e-3, 1e-2], and `weight_decay` in the continuous interval [1e-6, 1e-1] on a log scale.
- **EBM:** We tune `max_bins` in the integer interval [8, 512], `min_samples_leaf` and `max_leaves` in the integer interval [1, 50], `inner_bags` and `outer_bags` in the integer interval [1, 128], `learning_rate` in the continuous interval [1e-6, 100] on a log scale, and `max_rounds` in the integer interval [1000, 10000].
- **NodeGAM:** We tune `n_trees` in the integer interval [1, 256], `n_layers` and `depth` in the integer intervals [1, 4], `dropout` in the continuous interval [0, 0.9], `learning_rate` in the continuous interval [1e-3, 1e-2], and `weight_decay` in the continuous interval [1e-6, 1e-1] on a log scale.
- **GATSM:** We tune `nbm_batchnorm` in the descret set {False, True}, `nbm_dropout` in the continuous interval [0, 0.9], `attn_emb_size` in the integer interval [8, 128], `attn_n_heads` in the integer interval [1, 8], `attn_dropout` in the continuous interval [0, 0.9], `learning_rate` in the continuous interval [1e-3, 1e-2], and `weight_decay` in the continuous interval [1e-6, 1e-1] on a log scale. The optimal hyper-parameters for GATSM across all experimental datasets are provided in Table 6.

D Additional experiments

D.1 Inference speed

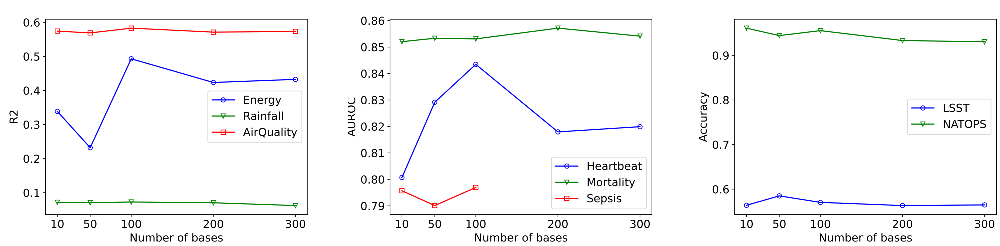
The inference speed of machine learning models is a crucial metric for real-world systems. We evaluate the throughput of various models. The results are presented in Table 7. Since the datasets have fewer features than the number of basis functions in NBM, NAM achieves higher throughput than NBM. Transparent tabular models typically exhibit fast speeds. However, their throughput significantly decreases in datasets with many features, such as Heartbeat, Mortality, and Sepsis, because they require the same number of feature functions as the number of input features. Transformer shows higher throughput than the transparent time series models because it does not require feature functions, which are the main bottleneck of transparent models. Additionally, the PyTorch implementation of Transformer uses the flash attention mechanism [61] to enhance its efficiency. NATM has slightly higher throughput than GATSM, as it does not require the attention mechanism and has fewer feature functions compared to the number of basis functions in GATSM.

Table 7: Inference throughput of different models.

	Energy	Rainfall	AirQuality	Heartbeat	Mortality	Sepsis	LSST	NATOPS
NAM	65.3K	1.8M	5.1M	139.1K	772.2K	23.9K	2.3M	147.9K
NBM	45.5K	1.1M	1.0M	55.9K	375.8K	6.5K	1.6M	85.6K
Transformer	30.9K	240.5K	174.2K	15.7K	161.9K	134.6K	214.4K	68.3K
NATM	5.3K	699.3K	241.3K	1.3K	N/A	N/A	28.6K	19.2K
GATSM	6.1K	350.6K	192.8K	1.2K	4.9K	3.8K	126.5K	12.5K

D.2 Number of basis functions

We evaluate GATSM by varying the number of basis functions in the time-sharing NBM. The results for forecasting, binary classification, and multi-class classification datasets are presented in Figure 6. For the Sepsis dataset, using 200 and 300 basis functions causes the out-of-memory error. For the Energy and Heartbeat datasets, performance improves up to 100 basis functions but shows no further benefit when the number of bases exceeds 100. In other datasets, performance changes are not significant with different numbers of basis functions. In addition, there is a trade-off between the number of basis functions and computational speed. Therefore, we recommend generally setting the number of basis functions to 100. Note that the performance of GATSM with this hyper-parameter depends on the dataset size and complexity. Hence, a larger number of basis functions may benefit more complex datasets.



(a) Forecasting (b) Binary (c) Multi-class

Figure 6: Performances of GATSM on the different number of basis functions.

E Additional visualizations

In addition to the interpretations on the AirQuality dataset in section 5.4, we present another interesting interpretations of GATSM on the Rainfall dataset.

Time-step importance: Figure 7 illustrates the average importance of all time steps at the final time step. The importance exhibit a cyclical pattern of rising and falling at regular intervals, indicating that GATSM effectively captures seasonal patterns in the Rainfall dataset.

Global feature contribution: Figure 8 illustrates the global behavior of features in the Rainfall dataset, with red bars indicating the density of training samples. Our findings indicate that low *Max Temperature* and high *Min Temperature* contribute to an increase in rainfall.

Local time-independent feature contribution: Figure 9 shows the local time-independent feature contributions. Consistent with the global interpretation, *Avg. Temperature* and *Min Temperature* have positive correlations with rainfall, while *Max Temperature* has a negative correlation with rainfall.

Local time-dependent feature contribution: Figure 10 shows the local time-dependent feature contributions. All features exhibit patterns similar to the local time-independent contributions. However, we found that *Avg. Temperature* and *Min Temperature* have time lags between feature values and contributions.

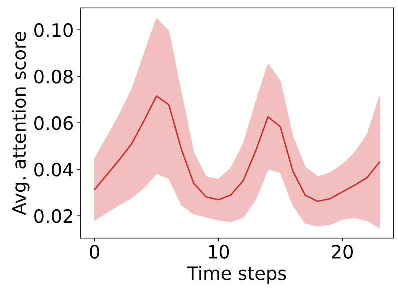


Figure 7: Average attention scores of time steps on the Rainfall dataset.

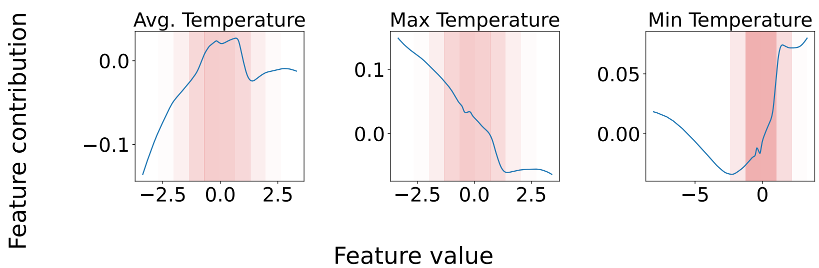


Figure 8: Global interpretations of features in the Rainfall dataset.

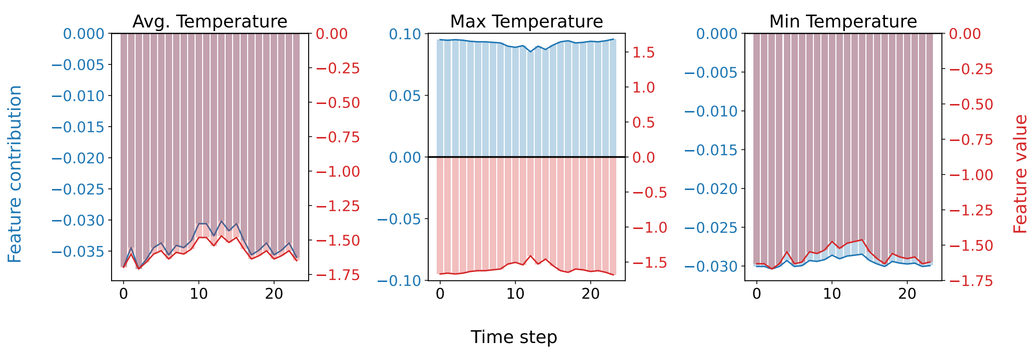


Figure 9: Local time-independent contributions of features in the Rainfall dataset.

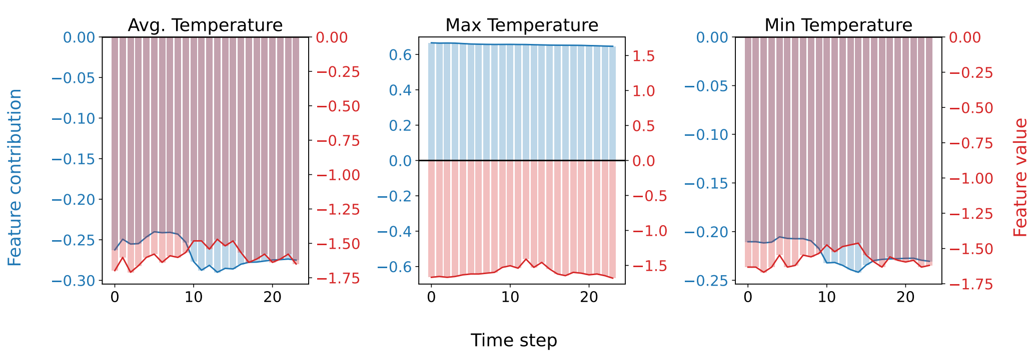


Figure 10: Local time-dependent contributions of features in the Rainfall dataset.