Spatio-Temporal Consistency enhanced Differential Network for Interpretable Indoor Temperature Prediction

Dekang Qi[1,2,3], Xiuwen Yi[2,3*], Chengjie Guo[4], Yanyong Huang[5]
Junbo Zhang[2,3,1], Tianrui Li[1], Yu Zheng[2,3,1]
[1]Southwest Jiaotong University, Chengdu, China, [2] JD iCity, JD Technology, Beijing, China
[3] JD Intelligent Cities Research, Beijing, China, [4]Xidian University, Xi'an, China
[5]Southwestern University of Finance and Economics, Chengdu, China
{dekangqi,msjunbozhang,msyuzheng}@outlook.com,xiuwenyi@foxmail.com
chengjie.guo@stu.xidian.edu.cn,huangyy@swufe.edu.cn,trli@swjtu.edu.cn

ABSTRACT

Indoor temperature prediction is crucial for decision-making in central heating systems. Beyond accuracy, predictions shall be interpretable, i.e. conform to the laws of physics; otherwise, it may lead to system failures or unsafe conditions. However, deep learning models often face criticism regarding interpretability, which limits their application in such settings. To this end, we propose a **S**patio-**T**emporal **Con**sistency enhanced Differential Network (**CONST**) for interpretable indoor temperature prediction. Our approach mainly consists of a differential predictive module and a spatio-temporal consistency module. Modeling the influential factors, the first module solves the issue of multicollinearity through the differential operation. Considering the heterogeneity of global and local data distribution, the second module characterizes the temporal and spatial consistency to mine the universal pattern by multi-task learning, thereby improving the prediction interpretability. Besides, we propose a set of interpretability metrics to overcome the drawbacks of partial dependence plot metric, which are more practical, zero-centered, flexible, and numerical. We conclude experiments on a real-world dataset with four heating stations. The results demonstrate the advantages of our approach over various baselines, where the interpretability can be improved by 8 times on cRPD while maintaining high accuracy. We deployed CONST on the SmartHeat system, providing fine-grained forecasts every hour for 13 heating staions in the Tianjin city.

CCS CONCEPTS

• **Applied computing** → **Forecasting**; • **Information systems** → **Spatial-temporal systems**.

KEYWORDS

Indoor Temperature, Interpretable Prediction, Spatio-temporal Consistency, Differential, Urban Computing

* Xiuwen Yi is the corresponding author.

KDD '24, August 25–29, 2024, Barcelona, Spain

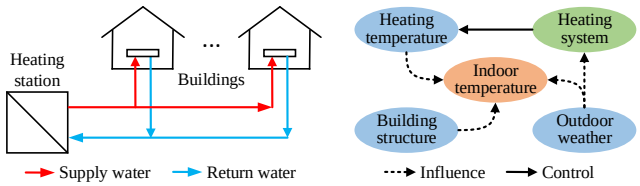
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Dekang Qi[1,2,3], Xiuwen Yi[2,3*], Chengjie Guo[4], Yanyong Huang[5] and Junbo Zhang[2,3,1], Tianrui Li[1], Yu Zheng[2,3,1]. 2018. Spatio-Temporal Consistency enhanced Differential Network for Interpretable Indoor Temperature Prediction. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (KDD '24).* ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

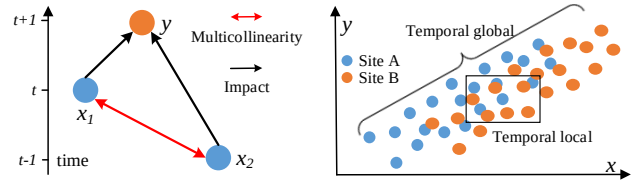
Indoor temperature has a great impact on living comfort for the citizens. To maintain a comfortable indoor temperature during winter, central heating systems are widely used in regions with medium to high latitudes, leveraging their benefits of energy efficiency, superior comfort, and user convenience [6, 16]. The schematic diagram of the heating system is shown in Fig. 1(a). Typically, a central heating station transports supply (hot) water to thousands of residential houses through pipes and continuously releases heat into the houses. As shown in Fig. 1(b), the indoor temperature is mainly affected by heating temperature, outdoor weather, and building structure. With better control of the heating system, the indoor temperature will be more stable, reducing uncomfortable experiences. Thus, it is essential to predict the indoor temperature, providing the control strategic-making support for heating dispatchers.

Safety is crucial in the central heating system, as improper control decisions can lead to system failures or unsafe conditions. Hence, prediction results shall be interpretable, in addition to being highly accurate. Interpretability refers to the degree to which a human can consistently predict the model's outcomes[23]. It usually means the predictions align with physical laws or common sense, e.g., Fourier's Law of Heat Conduction. For example, if other conditions remain unchanged and the predictions do not increase regardless of how much the heating temperature is increased, it is obviously unexplainable. It is imperative that the predictions adhere to the expected behaviors of heating systems, ensuring that control strategies are inherently safe and reliable.



(a) Central heating system (b) Influence relationship

Figure 1: Heating system and influence relationship.



(a) Complex impact of multiple factors (b) Heterogeneity of data distribution

Figure 2: The challenge of improving interpretability.

The prediction methods can be summarized into three categories. Physical rule-based methods, such as simulation-based Energyplus [5] and DOE-2 [26], are undoubtedly interpretable. However, they require hard-to-obtain building information, making large-scale deployment difficult. By simplifying the problem with only easy-obtained data for model training, simple machine learning methods (such as linear regression) [23] generally gain interpretability yet still sacrificing accuracy. Deep learning based methods [6, 21, 22, 30] are widely used for high accuracy. However, interpretability often faces criticism due to its complexity and opacity, which significantly limits its application in such settings.

To improve the interpretability of deep indoor temperature prediction models, several challenges need to be addressed:

The complex impact of multiple factors, especially for multi-collinearity. Indoor temperature is affected by multiple influential factors with different time delays. For example, cold wind penetration takes effect quickly, but the effect of heating temperature takes several hours to manifest. Moreover, there is multicollinearity among factors, which may cause difficulty with the reliability of the estimates of the model parameters [2]. For example, if $y = x_1 + x_2$ and $x_1 = 2x_2$, this is extreme multicollinearity. Here, x_1 and x_2 are independent variables, and y is a dependent variable. The relationship can take countless forms, for instance, $y = 0.5x_1 + 2x_2$, $y = -x_1 + 5x_2$, which diverges from the true underlying relationship, rendering them unable to be fully explanatory.

The heterogeneity of global and local data distribution masks the universal pattern. As shown in Figure 2(b), the global (entire samples) and local (nearby samples) data distributions are highly heterogeneous. Noise affects the local data distribution, leading to short-term disorder; however, they exhibit minimal influence on the global perception, indicating an approximate long-term linear trend. Obviously, it is easy to lose interpretability by relying solely on chaotic local data, while accuracy may be lost only by regular global data. Learning high-precision and interpretable models from heterogeneous data is very difficult, and it requires mining the universal patterns that are applicable in different times and spaces rather than being misled by one-sided patterns.

To address the above challenges, we propose the **S**patio-**T**emporal **Con**sistency enhanced Differential Network (**CONST**) for interpretable indoor temperature prediction. Our approach can significantly improve prediction interpretability while maintaining high accuracy without relying on physical rule-based or simple machine-learning models. Our contributions are listed below:

- We propose an interpretable CONST network, which can solve multicollinearity through differential operation and characterize temporal and spatial consistency to mine the universal pattern from heterogeneous data distribution by multi-task learning.
- We propose a set of interpretability metrics to overcome the drawbacks of existing metrics (Partial Dependence Plot), which are more practical, zero-centered, flexible, and numerical.
- We conducted experiments on a real-world dataset, and the results demonstrated the advantages of our approach, which can significantly improve interpretability by 8 times on cRPD.
- We deployed the proposed CONST into the SmartHeat system on the cloud, providing fine-grained forecasts every hour for 13 heating staions in the city of Tianjin, China.

2 OVERVIEW

2.1 Problem Formulation

Given the observation data X_{HT}^{S} from multiple stations, the objective is to predict future label data Y_{FT}^{S}. Among them, X is the historical data. $HT = \{t-n, ..., t\}$ is the historical time, where t is the current moment, and n is the total number of historical moments. Y is the future label data. $FT = \{t+1, ..., t+l\}$ is the future time, where l is the length of the prediction time window. $S = \{S_1, ..., S_h\}$ represents stations, where h is the amount of stations.

2.2 Overview

Figure 3 illustrates the framework of the Spatio-Temporal Consistency enhanced Differential Network (CONST), considering the data from indoor temperature, heating temperature, and outdoor weather. To improve the interpretability, CONST mainly consists of two modules: 1) Differential predictive module: complex influence of multiple factors is modeled through basic deep learning layers for feature extraction and fusion. Moreover, the multicollinearity is solved through the differential and restoration operations. 2) Spatio-temporal consistency module: the temporal and spatial consistency are characterized from heterogeneous global and local data distributions to mine the universal patterns by multi-task learning. Firstly, we sample the temporal global and local pairs to construct tasks with temporal discounting for weighting, using a fully shared strategy to learn temporal consistency. Secondly, we introduce the nearby spatial station to construct tasks, adopting a semi-sharing strategy to learn spatial consistency. Here, the output of the temporal local task for the target station is considered as a core prediction, while the others are regarded as auxiliary predictions.

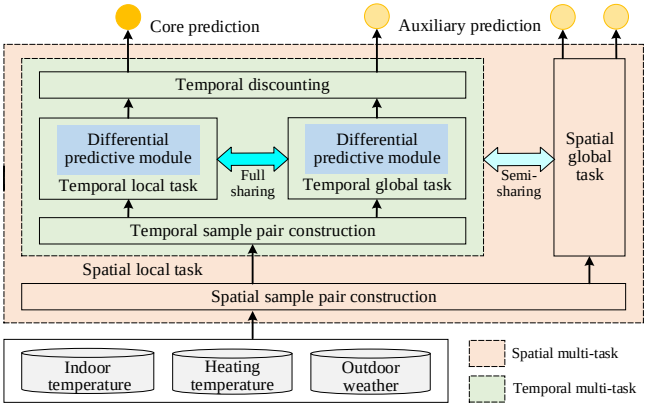


Figure 3: The framework of the CONST network.

3 METHODOLOGY

3.1 Differential predictive module

3.1.1 ***Basic deep learning layers.*** Indoor temperature is influenced by various factors, such as heating temperature and meteorological conditions (e.g., outdoor temperature, weather and wind). Furthermore, these factors have complex interactions. Thus, we use some basic deep-learning layers to model the influence.

Specifically, continuous features (heating temperature, outdoor temperature, wind speed, and sunlight intensity) are standardized and discrete features (wind direction, weather, day of week, and hour of day) are embedded. Each discrete feature is assigned an embedding layer, where the dimension is the total number of unique values. Subsequently, k moments of standardized continuous features and embedded discrete features are concatenated together to obtain the feature tensor. To learn the interactions, we use a stacked two-layer LSTM and a fully connected layer to fuse these features, which can be replaced with MLP, ResNet, and other models.

3.1.2 ***Differential & Restoration.*** Multicollinearity refers to the linear relation among two or more variables, which is usually evaluated using Variance Inflation Factor (VIF) [2]. If VIF is greater than 10, it indicates strong multicollinearity, while if VIF is close to 1, it indicates there is no issue of multicollinearity. We calculate VIFs of continuous features under different settings, which are shown in Table 1.

When fed features with raw values directly, the VIFs of heating temperature and historical indoor temperature exceed 142, indicating strong multicollinearity. Deep learning methods usually use historical labels as features, making them less interpretable. When using features without historical indoor temperature, the VIFs are all less than 4.5, and multicollinearity is significantly reduced. Physical rule-based models and simple machine learning models generally do not use historical labels as features, which is one reason for interpretability. However, such a process wastes the information of historical labels, resulting in lower accuracy. To deal with the multicollinearity problem, the differential operation is the simplest and most effective way. After the difference, the VIFs of all features are significantly reduced to less than 1.5. Thus, we adopt differential operation combined with deep learning layers.

Differential operation. By making the difference between the features at time t and the features at time τ, we can get the differential feature ΔX and the differential time Δt. In addition, the differential label Δy is also calculated. Notably, it is unreasonable to directly make differences in discrete features. It is necessary to make differences in the embedding space. Therefore, the differential operation is performed after feature concatenation.

$$\Delta X = X^t - X^\tau; \Delta t = t - \tau; \Delta y = y_{t+1} - y_{\tau+1} \tag{1}$$

Restoration operation. Due to the differential operation, only the predicted difference label $\Delta\hat{y}$ was obtained after ΔX passed through the fusion layers. Therefore, we adapt the restoration operation, which adds $\Delta\hat{y}$ to the historical label $y_{\tau+1}$ to get the desired predicted label $\hat{y}_{t+1}$, as shown in equation 2.

$$\hat{y}_{t+1} = y_{\tau+1} + \Delta\hat{y} \tag{2}$$

Table 1: Variance Inflation Factor

Feature	Raw value	W/o indoor temp.	Difference
Indoor temp.	148.79	/	1.42
Heating temp.	142.92	3.77	1.02
Outdoor temp.	2.13	1.33	1.33
Sunlight intensity	1.89	1.88	1.31
Wind speed	4.46	4.44	1.03

3.2 Spatio-temporal consistency module

In addition to multicollinearity, it is important to possess a universal pattern for improving interpretability. Specifically, the universal pattern can be suitable in different times and spaces. For example, Fourier's Law of Heat Conduction [18] can suit different unit times, e.g., one minute, one hour, one day, and different spatial stations. However, the heterogeneity of global and local data distribution makes it difficult. Therefore, we propose the spatio-temporal consistency module that utilizes the knowledge-sharing ability by multi-task learning, which can mine the universal pattern.

3.2.1 ***Temporal multi-task mechanism.*** We construct temporal global and local tasks to learn patterns separately and adapt the full sharing strategy to learn temporal consistency.

Temporal global and local tasks construction. We first construct global and local sample pairs through temporal multi-sampling. The characteristics of the data are captured through global and local tasks, where each task consists of a differential predictive module. Subsequently, the importance of the sample pair is adjusted through temporal discounting according to the time interval.

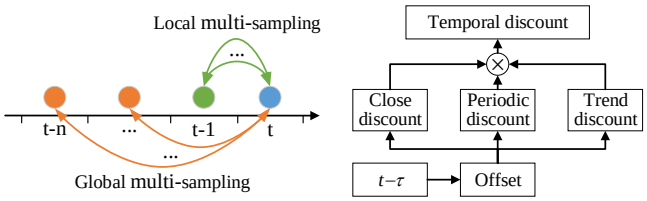
(1) Temporal sample pairs construction. In the temporal dimension, the key difference between global and local data lies in their time interval. Therefore, we employ sample pairs with different time intervals to reflect the characteristics of the data. As shown in Figure 4(a), we construct sample pairs through temporal multi-sampling. It includes global and local sampling, and multi-sampling.

Each sample pair includes a current sample $\{X_t, y_{t+1}\}$ and a historical sample $\{X_\tau, y_{\tau+1}\}$, making it convenient to process in differential operations. According to Equation 3, t is the current moment; τ is the historical moment. In the temporal local sample pair, $\tau = t-1$. In temporal global sample pair, $\tau = t'$, means arbitrary moment which is randomly selected before the current moment.

$$\tau = \begin{cases} t-1, & temporal\ local\ sampling \\ t', & temporal\ global\ sampling \end{cases} \tag{3}$$

In temporal global sampling, t' is randomly selected, and random errors may be introduced. Therefore, we repeat the temporal global sampling operation multiple times to reduce the impact of random errors. It can be considered as data enhancement. In addition, in order to correspond with multiple global sampling, local sampling is also repeated multiple times. Among them, the number of repetitions of temporal sampling is recorded as m_t. n is the number of historical moments. After temporal multi-sampling, $m_t \times n$ global sample pairs and $m_t \times n$ local sample pairs will be obtained.

(2) Temporal discounting. The temporal global sample pairs have long time intervals. The longer the time interval, the greater the



(a) Temporal multi-sampling (b) Temporal discounting

Figure 4: Temporal discounting and multi-sampling.

influence of unobservable factors, which will lead to greater cumulative errors. Inspired by the discounting in economics and reinforcement learning [14], we propose temporal discounting to deal with it and reduce the impact of cumulative errors.

Considering the temporal properties, we divide temporal discounting into close discounting, periodic discounting and trend discounting, as shown in Figure 4 (b). Close discounting indicates that data with closer time intervals has greater importance. Here, we use γ_c to represent the close discount coefficient and calculate n_c by the number of days in the interval, using $R_c = {\gamma_c}^{n_c}$ to calculate the close discount rate R_c. Similarly, trend discounting can be calculated. Periodic discounting indicates that the closer the position within the period, the greater its importance. We take 24 hours as a period, n_p is the distance within the period, which can be calculated according to Equation 6. When the time interval is 12 hours, the distance within the period is the furthest. Similarly, we can calculate the periodic discount rate R_p. Finally, the product of the three discount rates is used as the temporal discount rate. Here, the temporal discounting only applies to global sample pairs. As shown in Equation 7, we offset the time interval by 1 hour. Thus, the temporal discount rate R of local sample pairs is equal to 1, which will not affect local sample pairs.

$$R = R_c \times R_p \times R_t = \gamma_c^{n_c} \times \gamma_p^{n_p} \times \gamma_t^{n_t} \tag{4}$$

$$n_c = {\Delta t'}_{day}, \ n_t = {\Delta t'}_{week} \tag{5}$$

$$n_p = \begin{cases} {\Delta t'}_{hour}, & {\Delta t'}_{hour} < 12 \\ |{\Delta t'}_{hour} - 24|, & {\Delta t'}_{hour} \geq 12 \end{cases} \tag{6}$$

$$\Delta t' = \Delta t - 1 \tag{7}$$

Temporal discounting can be used as a correction method for the loss weight, applied between fusion layers and during the restoration operation. We multiply the temporal discount rate R by the predicted result $\Delta\hat{y}$, and the ground truth Δy is also multiplied by R. In other words, it can scale the loss according to the time intervals between sample pairs. As shown in the Equation 8, where n is the number of historical moments, m_t is the number of multi-sampling, and i represents the i-th sampling pair, $Loss_{TT}$ represents the loss of the temporal task. The larger the time interval between sample pairs, the more the loss is reduced and the lower the contribution to the sum of losses for all sample pairs.

$$Loss_{TT} = \frac{1}{m_t \times n} \sum_{i=1}^{m_t \times n} R_i \times |\Delta y_i - \Delta\hat{y}_i| \tag{8}$$

Full sharing strategy. Temporal local tasks can achieve high prediction accuracy, while local data are easily contaminated by noise, reducing the interpretability. Global data has stronger anti-noise performance, and temporal global tasks can achieve higher interpretability. Sharing parameters between two tasks can combine their advantages of them. However, due to the high heterogeneity in data distribution, all parameters must be shared, otherwise the model will be dominated by temporal local tasks and learn incorrect knowledge from messy local data. By sharing all parameters, the model can be forced to learn consistent patterns in both global and local data. Only patterns that are common in global and local data can be adopted by the model, and patterns that are only applicable to one kind of data will be discarded. Furthermore, sharing all parameters of two tasks does not mean there is a redundant task. Because the two tasks process different data respectively, the weights of the two tasks can be adjusted through the loss weight.

Here, we use $Loss_{TLT}$ to represent the loss of the temporal local task and $Loss_{TGT}$ to represent the loss of the temporal global task. They are different expressions of $Loss_{TT}$. We can balance between two tasks by loss weights α_1 and α_2. The loss of the temporal multi-task mechanism $Loss_{TMT}$ can be expressed as:

$$Loss_{TMT} = \alpha_1 Loss_{TLT} + \alpha_2 Loss_{TGT} \tag{9}$$

3.2.2 ***Spatial multi-task.*** We construct spatial global and local tasks to learn patterns separately and adapt the semi-sharing strategy to learn spatial consistency.

Spatial global and local tasks construction. We first obtain global and local samples through spatial multi-sampling. Then, characteristics are captured respectively through spatial global and local tasks, where each task comprises a temporal multi-task mechanism.

Spatial sample pairs construction. We sample the same number of spatial local and global samples. Among them, local samples are selected from target station data; global samples are sampled from all nearby stations. We record the repeat time of spatial sampling as m_s, which is also the ratio of the number of local samples to the number of local data. When $m_s = 1$, local data can be used as local samples, but not all global data has been sampled. We can increase m_s to reduce the number of unsampled data and avoid missing information. After spatial multi-sampling, the data is augmented and Equation 8 will be revised to:

$$Loss_{TT} = \frac{1}{m_t \times m_s \times n} \sum_{i=1}^{m_t \times m_s \times n} R_i \times |\Delta y_i - \Delta\hat{y}_i| \tag{10}$$

Semi-sharing strategy. The global and local data distributions in the spatial dimension are only slightly heterogeneous; there is no need to worry about learning contradictory knowledge from global and local tasks. Thus, sharing partial parameters is enough for the model to learn spatial consistency, called the semi-sharing strategy. The shared part deals with commonalities, and the non-shared part deals with characteristics. Shared network parameters include parameters of embedding layers and lower stack layers. Hyperparameters are usually shared but can also chosen differently for different tasks through experiments. For example, the time window length of feature (k) can reflect the influence of the building structure. The shorter the length of the pipe network, the better the thermal insulation performance, and the smaller the time delay of impacts, so a smaller k is more appropriate.

We use $Loss_{SLT}$ to represent the loss of spatial local task, and use $Loss_{SGT}$ to represent the loss of spatial global task. They are different expressions of $Loss_{TMT}$. Using β_1 and β_2 to represent the loss weight of the tasks. The loss of the spatial multi-task $Loss_{SMT}$, which is also the loss of the CONST model $Loss$, is:

$$Loss = Loss_{SMT} = \beta_1 Loss_{SLT} + \beta_2 Loss_{SGT} \tag{11}$$

4 METRICS OF INTERPRETABILITY

The Partial Dependence Plot (PD plot) is a commonly used and intuitive interpretability metric. However, it has some shortcomings: not conforming to the constraints of feature changes, inconvenient for comparison, and inconvenient for visualization with multiple features. Thus, we propose the Restricted Partial Dependence Plot and its variants to better measure the interpretability.

4.1 Restricted Partial Dependence

The Partial Dependence Plot [23] is drawn according to the partial dependence function. For a feature xv, it actually calculates the average value of prediction results when values of all samples' xv change to a specific value, while the values of other features remain unchanged. However, in practical applications, the value of features can usually be slightly increased or decreased based on their original value. Significantly changing the value of a feature in a short period may exceed the system's safety limits or become unreasonable data. For example, the change of outdoor temperature in one hour rarely exceeds 3 $^\circ C$.

Therefore, we calculate the average value of prediction results when values of the samples' feature xv increase or decrease in a restricting range based on their original values while the values of other features remain unchanged. We call it *Restricted Partial Dependence* (*RPD*), and the function is defined as Equation 12. Among them, xv is the varying feature; xo represents the other features. $\hat{f}$ is the trained model, n is the total number of samples, Δp represents times of the unit size, and t is the current moment. If the values of feature xv at time t to $t-k$ are changed simultaneously, it can be recorded as $RPD^{p}_{xv_{t\sim t-k}}$. When it is unnecessary to emphasize time, we simply record it as RPD^{p}_{xv}. For the continuous feature, the unit size can be determined according to specific scenarios. For example, 0.1 $^\circ C$ is used as the unit size for temperature. For the discrete feature, its value is an integer between $0 \sim Q$, and the unit size can be 1. RPD plot can be drawn according to the RPD function, where the vertical axis is the change of the dependent variable, and the horizontal axis is the change of the independent variable or the change multiple of the unit size. Compared with PD plot, RPD plot has some advantages. It is more consistent with actual security restrictions. It is zero-centred and focuses on the changing trends that are of concern in practical applications without being affected by prediction accuracy. When there are more than two features in xv, it can still be displayed graphically if p is the same.

$$RPD^{p}_{xv_t} = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(xv^i_t + \Delta p, xo^i_t) - \hat{f}(xv^i_t, xo^i_t) \tag{12}$$

4.2 Variants of Restricted Partial Dependence

PD plot and RPD plot are intuitive graphs and cannot provide numerical interpretability, which is inconvenient for comparison, so we proposed variants of RPD. If a highly interpretable model is selected as the benchmark, the RPD of the model is regarded as the ***Benchmark Restricted Partial Dependence*** (*bRPD*). Similar to *Accuracy* and *MAPE*, we define the ***Standardized Restricted Partial Dependence*** (*sRPD*) as shown in Equation 13, where p cannot be zero to avoid division by zero. Generally, there is only one feature in xv. When xv includes all features (F is the number of features), it is recorded as XV, and the *sRPD* at this point is called ***Model Restricted Partial Dependence*** (*mRPD*). If the features are considered to be independent and the value of each feature in XV can be changed sequentially rather than simultaneously, this is called ***Independent Model Restricted Partial Dependence*** (*imRPD*). If we only focus on the interpretability of a few core features XV', it is called ***Core Restricted Partial Dependence*** (*cRPD*). It is worth noting that the larger the value of the *sRPD*, *mRPD*, *imRPD*, and *cRPD*, the better.

$$sRPD^{-P\sim P}_{xv} = 100\% - \frac{1}{2P}\sum_{p=-P}^{P}\left|\frac{bRPD^{p}_{xv} - RPD^{p}_{xv}}{bRPD^{p}_{xv}}\right| \times 100\% \tag{13}$$

$$mRPD^{-P\sim P} = sRPD^{-P\sim P}_{XV} \tag{14}$$

$$imRPD^{-P\sim P} = \frac{1}{F}\sum_{j=1}^{F} sRPD^{-P\sim P}_{XV_j} \tag{15}$$

$$cRPD^{-P\sim P} = \frac{1}{F'}\sum_{j=1}^{F'} sRPD^{-P\sim P}_{XV'_j} \tag{16}$$

5 EXPERIMENT

5.1 Data sets

We conducted experiments on a real-world dataset with four heating stations in Tianjin city. The details is shown in Table 2. Among them, the time range of stations A and B is 2018/12/14~2019/03/15, and the time range of stations C and D is 2018/11/15~2019/03/15. The time interval of the data is 1 hour. These stations have 2192, 2176, 2704, and 2658 valid data, respectively.

5.2 Baselines

Traditional Models:

- **Multiple Linear Regression (Linear):** The multiple linear regression model is a classic simple machine learning model with high interpretability [23].

Table 2: Details of the Dataset

Station	A	B	C	D
Indoor temp./$^\circ C$	16.0~23.2	13.9~22.5	18.8~27.3	20.2~25.0
Heating temp./$^\circ C$	29.0~50.4	26.7~52.6	27.4~46.7	29.7~46.5
Outdoor temp./$^\circ C$	-10~18.4	-10~18.4	-11~17	-11~17
Sunlight /Lux	0~674	0~655	0~674	0~655
Wind speed /mps	0~6.7	0~6.7	0.1~12.3	0.1~12.3
Wind direction	9 types (e.g. no wind, east, southeast)			
Weather	16 types (e.g. sunny, rainy)			
Day of week	7 types (e.g. monday, tuesday)			
hour of day	24 types (e.g. 0, 1)			

- **Data Fitting Physical Model (Physical):** Refer to the method in the book [37], we derive a nonlinear model to predict indoor temperature based on physical rules and fit the parameters through real data. Please see appendix A.3 for details.

Classic Deep Learning Models:

- **MLP:** A simple deep neural network uses multi-layer fully connected layers named multilayer perceptron.
- **LSTM:** Long short-term memory network [12], suitable for processing time series with a long delay.
- **ResNet:** The Deep residual network [11] shines in fields such as computer vision. In this paper, the CNN layer is replaced by the fully connected layer.
- **MTL:** An extension of ResNet for time series prediction with multi-task learning [33]. Here, we replace the CNN layer with the fully connected layer and use splicing to replace the gating fusion component for discrete features.
- **TimesNet:** Transformer-based time series prediction approach, TimesNet [27] has achieved excellent prediction accuracy by introducing 2-dimensional space analysis.

CONST-based Models: The CONST network mainly relies on difference and spatio-temporal consistency, while the basic deep learning layers can be easily replaced with different structures. We obtain **CONST-MLP**, **CONST-ResNet**, **CONST-MTL**, and **CONST-LSTM** using MLP, ResNet, MTL, and LSTM as the stacked layers in CONST respectively.

5.3 Experimental Setup

5.3.1 ***Preprocessing.*** Continuous data is standardized as data with a mean of 0 and standard deviation of 1, and then de-standardized back to normal values during evaluation. We use 80% data for training and 20% data for testing. Then, 20% of the test data is used for validation. The experiment was repeated 3 times, and the mean value was taken as the final result.

5.3.2 ***Parameter Setting.*** The output dimension of each discrete feature embedding layer is set to 2. The $l2\_rate$ of the kernel regularizer in each stack layer is set to 0.01. A Dropout layer with a Dropout rate of 0.1 is used after the upper stack layers. α_1 , β_1, and m_s are set to 1. Other hyperparameters of all models in the same station remain consistent, and in different stations can be selected through experiments, as shown in Table 3. The MLP model uses 8 layers with 32-node fully connected layers stacked. The LSTM model uses 2 layers with 32-node LSTM layers stacked. Based on the MLP model, the ResNet model adds two shortcut connections. The MTL model uses the data of the last k (feature time window length) moments as close data, and uses the data of the previous day of each close data as the periodic data, and uses the data of the previous week of each close data as the trend data. In TimesNet, seq_len=k, label_len=$Ceiling(k/2)$, tok_k = 4. In the CONST-based models, the parameters of the basic deep learning layers are the same as those mentioned parameters of classic deep learning models.

5.3.3 ***Model Optimization.*** We use the Adam optimizer [15]. The learning rate is 0.0001, $beta_1 = 0.9$, $beta_2 = 0.999$, and the batch size is 64. Train for 300 epochs using an early stop strategy with a tolerance threshold of 10 rounds.

Table 3: Individualized hyperparameters for each station

Station	k	m_t	α_2	β_2	γ_c	γ_p	γ_t
A	4	3	1.1	1.15	0.9995	0.9982	0.9974
B	2	2	0.85	1.175	1	0.9982	0.9989
C	6	2	1.075	0.925	0.9991	0.9996	0.9989
D	7	2	0.975	1.05	0.9995	0.9992	0.9994

5.3.4 ***Experimental Environment.*** We use a cloud server with a 16-core CPU and 64G memory. Our model is constructed using the Keras library in Python, and multi-processing is used to speed up the computation. The code is shown in GitHub [1].

5.3.5 ***Evaluation Metrics.***

Accuracy:

$$Accuracy = 1 - MAPE \tag{17}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\% \tag{18}$$

Interpretability: We use the *PD Plot*, *RPD Plot*, and *cRPD* as described in Section 4 to evaluate interpretability.

5.4 Experimental results

5.4.1 ***Comparison with Different Baselines.*** We evaluate interpretability by calculating *cRPD* on the heating temperature and outdoor temperature. Here, we use the Data Fitting Physical Model as the benchmark of the interpretable model, so its *cRPD* is 100%. As shown in Table 4, the accuracy of the physical model is slightly higher than that of the linear model, indicating that detailed model derivation and nonlinear fitting have certain values, but are more cumbersome than the linear model. The linear model has a high interpretability of 95.10%. When applied to areas where it is inconvenient to build physical models, the linear model can be used as the benchmark of the interpretable model. Classic deep learning models have higher accuracy (above 98%), but their interpretability is lower than 11%. Among them, the TimesNet model mainly relies on the correlation between future and historical data for prediction. Paying too much attention to the correlation can improve accuracy, but it leads to low interpretability. Similarly, the interpretability of the MTL model is also relatively low. Due to our approach addressing the two challenges of interpretability and also making more comprehensive use of global and local data, compared with each basic deep learning model, the CONST-based models have achieved significant improvements in interpretability and small improvements in accuracy. The interpretability has increased by more than 4, 7, 14, and 8 times, and the error has decreased by 0.55, 0.37, 1.22, and 0.29, respectively. Since LSTM has the highest interpretability among basic deep learning models, CONST-LSTM has the highest interpretability among CONST-based models. The interpretability of CONST-LSTM has improved 8 times, from 10.42% to 92.50%, compared to the model with the best interpretability in classic deep learning models (LSTM). Compared with the model with the highest accuracy, it still maintains high accuracy, and the *accuracy* only decreased from 99.51% (TimesNet) to 99.41%.

[1] https://github.com/QiDekang/CONST

Table 4: Comparison with Different Baselines

Methods	Station A		Station B		Station C		Station D		Average		
	MAPE	*cRPD*(%)	*MAPE*	*cRPD*(%)	*MAPE*	*cRPD*(%)	*MAPE*	*cRPD*(%)	*MAPE*	*cRPD*(%)	*cRPD* ↑ (%)
Physical	6.57	100.00	5.45	100.00	6.74	100.00	7.35	100.00	6.53	100.00	-7.50
Linear	6.95	97.75	5.37	96.18	6.90	98.38	7.05	88.08	6.57	95.10	-2.60
TimesNet	0.62	0	0.43	0	0.59	0	0.32	0	0.49	0	**92.50**
MLP	1.21	7.68	0.53	2.26	1.01	8.83	1.02	14.62	0.94	8.35	**84.15**
CONST-MLP	0.41	38.24	0.51	43.74	0.39	31.65	0.24	49.31	0.39	40.73	**51.77**
ResNet	0.91	3.11	0.60	3.17	0.81	4.27	0.72	14.97	0.76	6.38	**86.12**
CONST-ResNet	0.40	50.04	0.49	45.41	0.43	58.43	0.25	41.71	0.39	48.90	**43.6**
MTL	3.84	0.18	0.47	0.55	2.00	1.94	0.62	5.70	1.73	2.09	**90.41**
CONST-MTL	0.45	41.64	0.50	0.00	0.44	37.14	0.24	45.58	0.41	31.09	**61.41**
LSTM	0.94	10.17	**0.77**	6.93	1.09	11.42	0.72	13.16	0.88	10.42	**82.08**
CONST-LSTM	**0.53**	**91.60**	0.96	**89.52**	**0.58**	**95.90**	**0.31**	**92.96**	**0.59**	**92.50**	-

Table 5: Ablation Experiment Results

Methods	Average	
	MAPE	*cRPD*(%)
CONST-w/o-SMT (spatial multi-task)	0.59	90.28
CONST-w/o-TMT (temporal multi-task)	**0.28**	4.90
CONST-w/o-DO (differential operation)	0.55	4.57
CONST-w/o-DF (discrete features)	0.48	84.41
CONST	0.59	**92.50**

5.4.2 ***Results of Ablation Experiment.*** As shown in Table 5: The interpretability of CONST-w/o-TMT and CONST-w/o-DO is poor, indicating that the temporal multi-task and the differential operation are indispensable. The interpretability of CONST-w/o-SMT is slightly lower, indicating that considering the commonalities of multiple stations can help improve the interpretability. Although removing discrete features will slightly improve the prediction accuracy, the interpretability will be reduced. It shows that discrete features are also valuable for improving interpretability.

5.4.3 ***Interpretability on PD Plot vs RPD Plot.*** Figure 5(a) illustrates comparing results of different models on the PD plot metric using the data in station A. The trend of the classic deep learning model (e.g., LSTM) is significantly different from that of the physical model, while the trend of the CONST based model (e.g., CONST-LSTM) is more similar to that of the physical model, proving that CONST can significantly improve the interpretability. The PD plot is easily affected by prediction accuracy, while the results on RPD plot shown in Figure 5 (b) is zero-centered, and the above trend is more pronounced. According to the intersection points of the red straight line ($\Delta y = 0.2$) and the curves in the figure, when increasing the indoor temperature in the future by 0.2 $^\circ C$, the changes in the heating temperature obtained by the physical model, linear model, and CONST-LSTM model are similar. Both are about 1 $^\circ C$. However, the classic deep learning model has no intersection with the red straight line. If we want to change the indoor temperature, we need to provide a heating adjustment that is far beyond common sense and system security restrictions. It is even possible to obtain the direction of adjustment opposite to the physical model.

5.4.4 ***Long-term Prediction.*** As shown in Figure 6(a), when we predict the long-term indoor temperature, the prediction error increases. The *cRPD* of CONST-LSTM decreases slightly, but remains at a high level. The *cRPD* of LSTM slightly increases.

5.4.5 ***The interpretability of different features.*** As shown in Figure 6(b), in the CONST-LSTM model, compared to changing the outdoor temperature, changing the heating temperature has a more significant impact on the indoor temperature.

5.4.6 ***Parameter Experiments.***

Time window length of feature (k). As shown in Figure 7(a), when k is increased, the interpretability and error present a trend of first increasing and then decreasing. The influence of various factors on indoor temperature has time lags, and as k increases, the data of reaction hysteresis can be gradually obtained, and the interpretability increases. When k is too large, and the redundant data also increases, so the interpretability gradually decreases. The time lags of stations are different, and the interpretability can be improved by setting individualized k.

Loss weight. As shown in Figure 7(b), as α_2 increases, the weight of the global task increases, so the interpretability increases, and the accuracy decreases. The result for β_2 is similar.

Data enhancement. The local task $\tau = t-1$ is a special case among the many values of $\tau = t'$ in the global task. The greater the m_t, the higher the weight of the local task, making the interpretability decline and the accuracy increase, as shown in Figure 7(c). The result for m_s is similar.

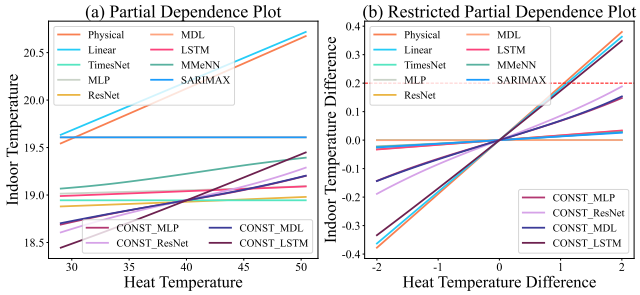


Figure 5: Interpretability on PD Plot vs RPD plot

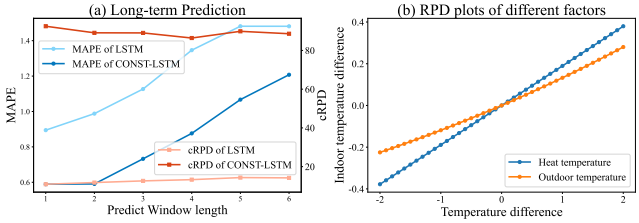


Figure 6: Predict the length and interpretability of features

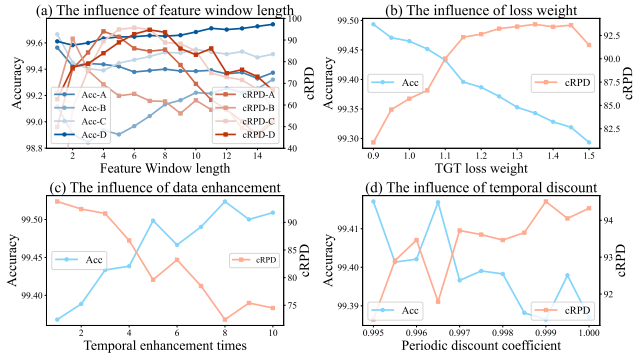


Figure 7: Experimental results with different parameters

Temporal discount. Temporal discount adjusts the importance of global data, In Figure 7(d), as γ_p increases, the interpretability increases, and the accuracy decreases. γ_c and γ_t yield similar results.

6 SYSTEM DEPLOYMENT

We deployed the proposed CONST approach into the SmartHeat system on the cloud. The system mainly consists of four modules. The data collector module continuously collects real-time data, such as heating data, indoor temperature data, and weather data. Considering the limited coverage and data missing issues of indoor temperature, the data supplement module fills the missing values based on their spatio-temporal neighbors. Then, the prediction module predicts future indoor temperatures for the next time interval. After that, the control module provides the recommended heating temperature using optimization and control strategies.

The system offers a user-friendly web interface to the heating dispatcher. As shown in Figure 8, it is mainly comprised of three components on the prediction module. The upper component displays the overview of one selected heating station. The button left component focuses on prediction accuracy, providing a comparison between predicted indoor temperature and actual indoor temperature over different time spans. The button right component focuses on the prediction interpretability, displaying the interpretability of core influential factors on the *cRPD* metric.

Taking into account the frequency of data collector, the predictive system is designed to execute hourly updates, generating forecasts for the subsequent six-hour period for the 13 heating stations in Tianjin, China. To ensure continued accuracy, the CONST model is retrained monthly, utilizing the latest heating data inputs. Such proactive model recalibration is critical for capturing and adapting to dynamic patterns and shifts.

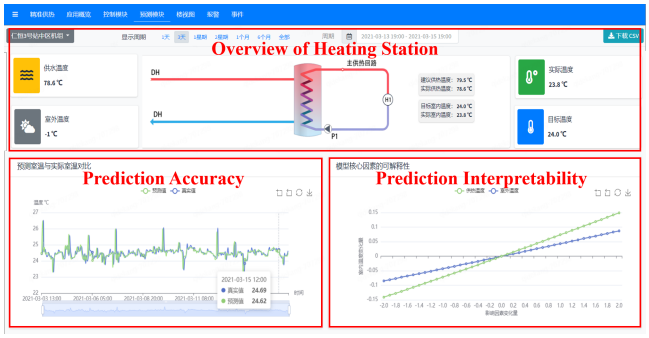


Figure 8: The Web interface of SmartHeat System

7 RELATED WORK

Indoor Temperature Prediction: There are three types of methods. (1) Simulation models such as Energyplus [5] and DOE-2 [26] are typical physical-based models, which require detailed information about buildings and are not convenient for large-scale deployment. (2) Data-driven methods, especially deep learning methods, have high accuracy but low interpretability. Such as papers based on NNARX (Autoregressive neural networks with exogenous variables) [1, 7, 21, 22], LSTM and seq2seq [9], and LSTM and CNN [8]. (3) The RC thermal network [3, 10, 13] is a typical hybrid model, which uses a network composed of resistors and capacitors to simplify the physical model, and fits parameters through actual data. The demand for domain knowledge and hard-to-obtain data is still high.

Interpretable AI: The methods to imporve interpretability can be summarized into two types. One type is combines deep learning models with physical models [4, 29] or simple machine learning models (e.g., linear regression) [34], to balance accuracy and interpretability. However, the disadvantage lies in requiring hard-to-obtain data or difficulty in determining which factors should be included in the primary linear model. Our model is significantly different, which does not rely on physical and simple machine learning models, and can plug and play to improve the interpretability of deep learning models. Another type is utilizing the similarity between time series segments and basic shapes [17] or historical segments [28], which focus on the dependent variable, while we focus on the influence of the independent variable on the dependent variable.

Deep Learning for Spatio-temporal Data Mining: There are many works that use deep learning methods to mine knowledge from massive spatio-temporal data[25]. Various deep learning models, such as CNN-based [24], ResNet-based[32], Transformer-based [19, 36], are proposed to learns spatiotemporal dependencies simultaneously, achieving high prediction accuracy. Additionally, multi-task learning (MTL) aims to leverage useful information contained in multiple related tasks to help improve the generalization performance of all the tasks [35]. The MTL-based [20, 32] methods have achieved high accuracy in multiple fields. Unlike them, we use a fully shared and semi-sharing strategy simultaneously.

8 CONCLUSION

In this paper, we propose a CONST network for indoor temperature prediction for central heating systems. Our approach can directly improve the interpretability of deep learning models without relying on physical rule-based models and simple machine learning models. CONST solves multicollinearity through differential operation and characterizes temporal and spatial consistency to learn universal patterns from heterogeneous data distribution by multi-task learning. The experimental results on a real-world dataset with four heating stations demonstrate that our model can significantly improve interpretability by 8 times on *cRPD* while maintaining a high prediction accuracy. In addition, we have deployed CONST on the SmartHeat system in Tianjin, China, providing fine-grained forecasts every hour for 13 heating stations.

ACKNOWLEDGMENTS

This paper is supported by the National Key R&D Program of China (2023YFC2308703) and Beijing Nova program (Z211100002121119).

REFERENCES

[1] Zakia Afroz, Tania Urmee, GM Shafiullah, and Gary Higgins. 2018. Real-time prediction model for indoor temperature in a commercial building. *Applied energy* 231 (2018), 29–53.
[2] Aylin Alin. 2010. Multicollinearity. *Wiley interdisciplinary reviews: computational statistics* 2, 3 (2010), 370–374.
[3] Thomas Berthou, Pascal Stabat, Raphael Salvazet, and Dominique Marchio. 2014. Development and validation of a gray box model to predict thermal behavior of occupied office buildings. *Energy and Buildings* 74 (2014), 91–100.
[4] Yuntian Chen and Dongxiao Zhang. 2021. Theory-guided deep-learning for electrical load forecasting (TgDLF) via ensemble long short-term memory. *Advances in Applied Energy* 1 (2021), 100004.
[5] Drury B Crawley, Linda K Lawrie, Curtis O Pedersen, and Frederick C Winkelmann. 2000. Energy plus: energy simulation program. *ASHRAE journal* 42, 4 (2000), 49–56.
[6] Benoit Delcroix, Jérôme Le Ny, Michel Bernier, Muhammad Azam, Bingrui Qu, and Jean-Simon Venne. 2021. Autoregressive neural networks with exogenous variables for indoor temperature prediction in buildings. In *Building Simulation*, Vol. 14. Springer, 165–178.
[7] Benoit Delcroix, Jérôme Le Ny, Michel Bernier, Muhammad Azam, Bingrui Qu, and Jean-Simon Venne. 2021. Autoregressive neural networks with exogenous variables for indoor temperature prediction in buildings. In *Building Simulation*, Vol. 14. Springer, 165–178.
[8] Furkan Elmaz, Reinout Eyckerman, Wim Casteels, Steven Latré, and Peter Hellinckx. 2021. CNN-LSTM architecture for predictive indoor temperature modeling. *Building and Environment* 206 (2021), 108327.
[9] Zhen Fang, Nicolas Crimier, Lisa Scanu, Alphanie Midelet, Amr Alyafi, and Benoit Delinchant. 2021. Multi-zone indoor temperature prediction with LSTM-based sequence to sequence model. *Energy and Buildings* 245 (2021), 111053.
[10] MM Gouda, Sean Danaher, and CP Underwood. 2000. Low-order model for the simulation of a building and its heating system. *Building Services Engineering Research and Technology* 21, 3 (2000), 199–208.
[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[13] Ying Ji, Peng Xu, Pengfei Duan, and Xing Lu. 2016. Estimating hourly cooling load in commercial buildings using a thermal network model and electricity submetering data. *Applied Energy* 169 (2016), 309–323.
[14] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research* 4 (1996), 237–285.
[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[16] Jan F Kreider. 2000. *Handbook of heating, ventilation, and air conditioning*. CRC press.
[17] Guozhong Li, Byron Choi, Jianliang Xu, Sourav S Bhowmick, Kwok-Pan Chun, and Grace Lai-Hung Wong. 2020. Efficient shapelet discovery for time series classification. *IEEE transactions on knowledge and data engineering* 34, 3 (2020), 1149–1163.
[18] John H Lienhard. 2005. *A heat transfer textbook*. Phlogistron.
[19] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. 2022. Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting. (2022).
[20] Ye Liu, Yu Zheng, Yuxuan Liang, Shuming Liu, and David S Rosenblum. 2016. Urban water quality prediction based on multi-task multi-view learning. In *Proceedings of the 25th international joint conference on artificial intelligence*.
[21] Tao Lu and Martti Viljanen. 2009. Prediction of indoor temperature and relative humidity using neural network models: model comparison. *Neural Computing and Applications* 18 (2009), 345–357.
[22] A Mechaqrane and Mohcine Zouak. 2004. A comparison of linear and neural network ARX models applied to a prediction of the indoor temperature of a building. *Neural Computing & Applications* 13 (2004), 32–37.
[23] Christoph Molnar. 2020. *Interpretable machine learning*. Lulu. com.
[24] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems* 28 (2015).
[25] Senzhang Wang, Jiannong Cao, and Philip Yu. 2020. Deep learning for spatio-temporal data mining: A survey. *IEEE transactions on knowledge and data engineering* (2020).
[26] Frederick C Winkelmann and Stephen Selkowitz. 1985. Daylighting simulation in the DOE-2 building energy analysis program. *Energy and Buildings* 8, 4 (1985), 271–286.
[27] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *International Conference on Learning Representations*.
[28] Haixu Wu, Hang Zhou, Mingsheng Long, and Jianmin Wang. 2023. Interpretable weather forecasting for worldwide stations with a unified deep model. *Nature Machine Intelligence* (2023), 1–10.
[29] Ziyu Xiang, Mingzhou Fan, Guillermo Vázquez Tovar, William Trehern, Byung-Jun Yoon, Xiaofeng Qian, Raymundo Arroyave, and Xiaoning Qian. 2021. Physics-constrained automatic feature engineering for predictive modeling in materials science. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 10414–10421.
[30] Chengliang Xu, Huanxin Chen, Jiangyu Wang, Yabin Guo, and Yue Yuan. 2019. Improving prediction performance for indoor temperature in public buildings based on a novel deep learning method. *Building and Environment* 148 (2019), 128–135.
[31] Zhaohui Liu Yi Li, Yiwen Jian. 2015. Climate compensator adjust and impact analysis of the simulation. In *Proceedings of the Symposium on Heating Engineering Construction and Efficient Operation*. Gas& Heat, 195–200.
[32] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Thirty-first AAAI conference on artificial intelligence*.
[33] Junbo Zhang, Yu Zheng, Junkai Sun, and Dekang Qi. 2019. Flow prediction in spatio-temporal networks based on multitask deep learning. *IEEE Transactions on Knowledge and Data Engineering* 32, 3 (2019), 468–478.
[34] Xianli Zhang, Buyue Qian, Shilei Cao, Yang Li, Hang Chen, Yefeng Zheng, and Ian Davidson. 2020. INPREM: An interpretable and trustworthy predictive model for healthcare. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 450–460.
[35] Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering* (2021).
[36] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, Vol. 35. AAAI Press, 11106–11115.
[37] Pinghua Zou. 2018. *Heating engineering*. China Architecture & Building Press.

A APPENDICES

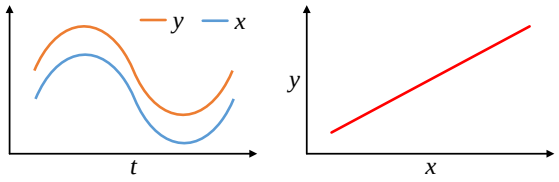
A.1 Terminology

Time perspective. In time series data, we use x to represent a feature and y to represent the label. As shown in Figure 9(a), taking time as the horizontal axis and displaying x and y data simultaneously in the figure is the time perspective. In this perspective, the main focus is on the patterns of data changes over time, such as temporal closeness, period, and trend [32]. These patterns should be classified as indirect effects. The typical approach is to refer to the data near (usually within a few moments) the current moment as short-term data; Data that is far from (usually within dozens of moments) the current moment is referred to as long-term data.

Feature perspective. As shown in Figure 9(b), using independent variable x as the horizontal axis and dependent variable y as the vertical axis represents the feature perspective. In this perspective, the main focus is on the laws of y changing with x, which should be classified as direct effects. Obviously, it is easier to obtain interpretability from direct effects.

Global and local in temporal dimension. In the temporal dimension, from the feature perspective, it can be divided into global data distribution and local data distribution. Local refers to data near the current moment, which is similar to the short-term comparison in the time perspective. Global refers to the entire data set, which may involve data with very large time intervals. It is similar to the ultra-long-term data from the time perspective.

Global and local in spatial dimension. In the spatial dimension, the distribution of data at multiple spatial locations is regarded as global data distribution; the distribution of data at a single location is regarded as local data distribution.



(a) Time perspective (b) Feature perspective

Figure 9: Different perspectives in the temporal dimension.

A.2 Another perspective on related work

Related works can be divided into three categories, and their relationships are shown in Figure 10. We use $f'()$ represents the constructed model and Y' represents the predicted output. The black box model mainly focuses on Y', and does not care about the exact form of $f'()$. The deep learning based model [6, 21, 22, 30] fall into this category, they usually have high prediction accuracy but low interpretability. The white-box model mainly focuses on $f'()$ and is interested in understanding the way that Y is affected as $x_1, ..., x_p$ change. Physical rule-based models [5, 26] and simple machine learning models (such as linear regression, decision rules) [23] both fall into this category, which generally gain interpretability by simplifying the problem. They are more interpretable, but difficult to apply or have lower accuracy. The gray-box model try to trade off accuracy and interpretability [4, 13, 29, 34]. They typically

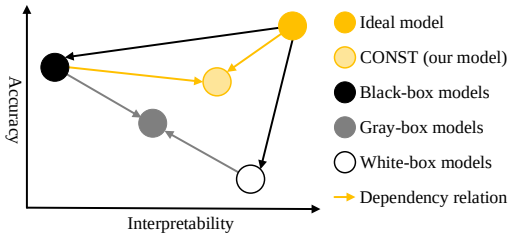


Figure 10: Method classification.

rely on the white box model to obtain interpretability. The disadvantages of the white box model are also inherited. Our model is significantly different from them. It does not rely on the white-box model (such as linear model, physical model, etc., which obtain interpretability by restricting the complexity of the model), and directly making the deep learning model more interpretable.

A.3 Data Fitting Physical Model for Indoor Temperature Prediction

The initial value setting of the nonlinear parameter estimation has a great influence on the performance. The method in book [37] can reduce a large number of parameters by comparing the design state with the actual state, which simplifies the setting of the initial value. In the case of only considering the main features (heating temperature, outdoor temperature), we derived a nonlinear model through physical laws, and fitted the nonlinear parameters through real data to obtain the data fitting physical model for indoor temperature prediction.

A.3.1 ***Model Derivation.*** Equation 19 ~ 22 are similar to Equation 12-1, 12-2, 12-5, and 12-4 in the book [37]:

$$Q_1 = Q_2 = Q_3 \tag{19}$$

$$Q_1 = q_v V(t_i - t_o) \tag{20}$$

$$Q_2 = aF\left(\frac{t_h + t_r}{2} - t_i\right)^{1+b} \tag{21}$$

$$Q_3 = Gc(t_h - t_r) \tag{22}$$

Among them, Q_1, Q_2, Q_3 are respectively the heat load of the building, the heat dissipation of the heat dissipation equipment, and the heat transported by the heating pipe network. q_v is the heating volume heat index of the building. V is the external volume of the building. t_o is the outdoor temperature; t_h is the supply water temperature, and t_r is the return water temperature. a and b are related parameters of heat transfer coefficient of radiator. F is the cooling area of the radiator. G is water flow. c is the specific heat capacity of water. The difference between Equations 20 and 21 in this paper and Equations 12-2 and 12-5 in the book [37] is that we use the actual indoor temperature t_i instead of the calculating indoor temperature $t_i^{'}$.

For design conditions, Equation 23~ 26 are the same as Equation 12-6~12-9 in book [37], where the superscript represents the design value:

$$Q_1^{'} = Q_2^{'} = Q_3^{'} \tag{23}$$

$$Q_1^{'} = q_v V(t_i^{'} - t_o^{'}) + + \tag{24}$$

$$Q_2^{'} = aF\left(\frac{t_h^{'}+t_r^{'}}{2} - t_i^{'}\right)^{1+b} \tag{25}$$

$$Q_3^{'} = G^{'}c(t_h^{'} - t_r^{'}) \tag{26}$$

The same as the Equation 12-12 in the book [37], comparing the two sets of equations of the actual situation and the design situation, we can get Equation 27:

$$\frac{t_i - t_o}{t_i^{'} - t_o^{'}} = \frac{\left(\frac{t_h+t_r}{2} - t_i\right)^{1+b}}{\left(\frac{t_h^{'}+t_r^{'}}{2} - t_i^{'}\right)^{1+b}} = \bar{G}\frac{t_h - t_r}{t_h^{'} - t_r^{'}} \tag{27}$$

Among them, $\bar{G}$ is the ratio of actual flow to design flow. Equation 28 can be deduced from the second and third terms in Equation 27:

$$t_i = \frac{t_h + t_r}{2} - \left[\bar{G}\frac{\left(\frac{t_h^{'}+t_r^{'}}{2} - t_i^{'}\right)^{1+b}}{t_h^{'} - t_r^{'}}(t_h - t_r)\right]^{\frac{1}{1+b}} \tag{28}$$

Adding correction coefficients λ_1, μ_1, ϵ_1, we can be obtained method 1 as Equation 29:

$$t_i^1 = \lambda_1\frac{t_h + t_r}{2} - \mu_1\left[\bar{G}\frac{\left(\frac{t_h^{'}+t_r^{'}}{2} - t_i^{'}\right)^{1+b}}{t_h^{'} - t_r^{'}}(t_h - t_r)\right]^{\frac{1}{1+b}} + \epsilon_1 \tag{29}$$

The multicollinearity between heating temperature and return temperature is too strong, which makes it easy to get un trustworthy coefficients when fitting parameters. The return temperature changes with the heating temperature, and the following linear regression can be used to fit the return temperature through the heat temperature. As shown in Equation 30, using the linear regression without intercept can avoid the situation of $t_h - t_r < 0$.

$$t_r = \rho \times t_h \tag{30}$$

Therefore, Equation 29 is revised to Equation: 31

$$t_i^1 = \frac{1+\rho}{2}\lambda_1 t_h - \mu_1\left[\bar{G}\frac{\left(\frac{t_h^{'}+t_r^{'}}{2} - t_i^{'}\right)^{1+b}}{t_h^{'} - t_r^{'}}(1-\rho)t_h\right]^{\frac{1}{1+b}} + \epsilon_1 \tag{31}$$

Equation 32 can be deduced from the first and third terms in the Equation 27:

$$t_i = \bar{G}\frac{t_i^{'} - t_o^{'}}{t_h^{'} - t_r^{'}}(t_h - t_r) + t_o \tag{32}$$

Adding correction coefficients λ_2, μ_2, ϵ_2, we can be obtained method 2 as Equation 33:

$$t_i^2 = \lambda_2\bar{G}\frac{t_i^{'} - t_o^{'}}{t_h^{'} - t_r^{'}}(t_h - t_r) + \mu_2 t_o + \epsilon_2 \tag{33}$$

According to Equation 30, Equation 33 is revised to Equation 34

$$t_i^2 = \lambda_2\bar{G}\frac{t_i^{'} - t_o^{'}}{t_h^{'} - t_r^{'}}(1-\rho)t_h + \mu_2 t_o + \epsilon_2 \tag{34}$$

Table 6: Parameters of physical model by nonlinear fitting

Station	λ_1	μ_1	ϵ_1	λ_2	μ_2	ϵ_2	ρ	w
A	0.270	1.056	16.62	6.548	0.551	3.423	0.929	0.757
B	0.678	1.013	11.56	0.677	0.561	6.078	0.868	0.565
C	0.448	1.168	85.22	5.070	0.513	-198.3	0.831	0.757
D	0.406	1.031	34.97	4.678	0.337	-35.29	0.839	0.673

Combining the method 1 and the method 2, the physical model can be obtained as Equation 35:

$$t_i = w \times t_i^1 + (1-w) \times t_i^2 \tag{35}$$

A.3.2 *Parameter setting and data fitting.*

Parameter settings: According to book [37] and article [31], the parameters are set as follows: $t_i^{'} = 18, t_o^{'} = -7, t_h^{'} = 75, t_r^{'} = 50, b = 0.16, \bar{G} = 1$.

Data Fitting: The derived physical model is nonlinear. We firstly fit the value of k through linear fitting, and then we use the lsqcurvefit method in Matlab to perform nonlinear fitting based on real data. We set the initial values of λ_1, μ_1, λ_2, and μ_2 to 1, and the value range is greater than or equal to 0. The initial values of ϵ_1 and ϵ_2 are set to 0, and the range is not limited. The initial value of w is set to 0.5, and the range is set to [0,1].

Fitting parameter results: The final fitting parameter results are shown in the table 6: