Learning Structured Components: Towards Modular and Interpretable Multivariate Time Series Forecasting

Jinliang Deng, Xiusi Chen, Renhe Jiang, Du Yin, Yi Yang, Xuan Song, and Ivor W. Tsang, *Fellow, IEEE*

Abstract—Multivariate time-series (MTS) forecasting is a paramount and fundamental problem in many real-world applications. The core issue in MTS forecasting is how to effectively model complex spatial-temporal patterns. In this paper, we develop a modular and interpretable forecasting framework, which seeks to individually model each component of the spatial-temporal patterns. We name this framework SCNN, as an acronym of Structured Component-based Neural Network. SCNN works with a pre-defined generative process of MTS, which arithmetically characterizes the latent structure of the spatial-temporal patterns. In line with its reverse process, SCNN decouples MTS data into structured and heterogeneous components and then respectively extrapolates the evolution of these components, the dynamics of which are more traceable and predictable than the original MTS. Extensive experiments are conducted to demonstrate that SCNN can achieve superior performance over state-of-the-art models on three real-world datasets. Additionally, we examine SCNN with different configurations and perform in-depth analyses of the properties of SCNN.

Index Terms—Spatial-temporal Data Mining, Time Series Forecasting, Deep Learning, Normalization.

1 Introduction

Multivariate time series (MTS) forecasting is a fundamental problem in the machine learning field [1], [2]. In the era of big data, massive promising applications can be formulated as MTS forecasting problems; examples include anticipating activities and events [3], nowcasting precipitation [4], forecasting traffic [2], and predicting pedestrian and vehicle trajectories [5]. The core issue in MTS forecasting is effectively capturing spatial-temporal patterns from MTS data. Here, spatial characteristics result from spatially external factors, such as regional population, functionality, and geographical location; temporal characteristics are yielded by temporally external factors, such as time of the day, day of the week, and weather conditions.

Conventional approaches are built upon a critical assumption that the time series to be modeled is stationary [6].

- *J. Deng is with Australian Artificial Intelligence Institute, University of Technology Sydney, Sydney, Australia. J. Deng is also affiliated with Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. E-mail: jinliang.deng@student.uts.edu.au.*
- *X. Chen is with University of California, Los Angeles, USA. Email: xchen@cs.ucla.edu.*
- *R. Jiang is with Center for Spatial Information Science, University of Tokyo, Tokyo, Japan. Email: jiangrh@csis.u-tokyo.ac.jp.*
- *D. Yin is with the Department of Computer Science and Engineering, University of New South Wales, Sydney, Australia. Email: yind7@outlook.com.*
- *Y. Yang is with Tencent WXG, Guangzhou, China. Email: paulyyyang@tencent.com.*
- *X. Song is with SUSTech-UTokyo Joint Research Center on Super Smart City, Department of Computer Science and Engineering, Southern University of Science and Technology (SUSTech), Shenzhen, China. Email: songx@sustech.edu.cn.*
- *Ivor W. Tsang is with the Center for Frontier AI Research, Agency for Science, Technology and Research (A*STAR), Singapore. Email: ivor.tsang@gmail.com.*

Xuan Song and Ivor W. Tsang are corresponding authors.

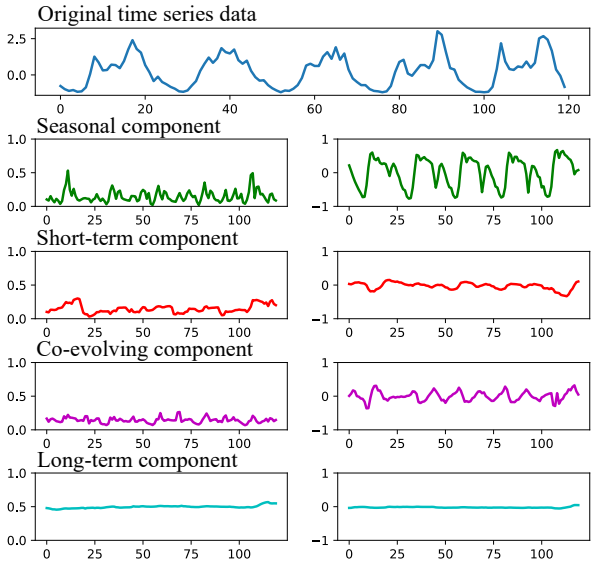


Fig. 1: Structured components extracted by SCNN from BikeNYC time series data. The underlying structure of TS might be far more complicated than just trend (long-term) and seasonal components.

However, real-world multivariate time series tend to be non-stationary, since they contain diverse structured patterns, such as continuity (which is at multiple resolutions), as well as seasonality. These structured patterns substantially complicate the dynamics of time series, imposing a tremendous challenge on the forecast. To achieve high accuracy, a forecast model must trace and project those complex dynamics resulting in diverse structured patterns. Most recent advanced methods are empowered by deep neural

networks, such as Transformers, temporal convolution networks (TCNs) and recurrent neural networks (RNNs). The upside of these methods is that they are not constrained by the assumption of stationarity [7], [8], [9], [10], [11], [12]. However, a common downside of these methods is that **they make the prediction process an entirely black box lacking interpretation for the internal mechanism.**

Time series decomposition [6] is a classical idea that decouples time series into trend, seasonal, and residual components. Recent works [11], [13], [14], [15], [16] marry this idea with deep learning architectures to achieve interpretability in time series forecasting tasks. Despite the improvement, these studies still have notable shortcomings. Particularly, **they have narrowly focused on long-term trend and seasonal components while ignoring the intricate, fine-grained structured components that can significantly affect the MTS forecasting**, as illustrated in Fig. 1. As a result, they usually fall short in providing a thorough and comprehensive understanding of the MTS dynamics.

In order to reach the goal of modeling the dynamics of MTS in an interpretable and modular fashion, our study proposes to characterize the structure of MTS from the first principle with a generative process and accordingly craft a structured component-based neural network (SCNN) for MTS forecasting. In particular, SCNN explicitly decouples the family of structured components that profile the dynamics of MTS data from different views, where each component is more traceable and predictable than the original hybrid data. More importantly, these components complement each other yet are sufficient to recover the original MTS. By decoupling these components in a deep and iterative fashion, richer types of structural information beyond the long-term and seasonal components can be extracted. Then, SCNN adopts a divide-and-conquer strategy to solve the projection of time series by customizing a simple law or a parameterized model for each structured component, in line with the nature of its dynamics. Finally, for the purpose of enhancing the robustness of SCNN, we construct auxiliary structural regularization in addition to the commonly used regression loss, steering the model to pay more attention to the structured components that are less susceptible to corruption.

In a nutshell, we demonstrate an MTS forecasting solution that is modular, intuitive and promising. Each operation is deployed to handle particular dynamics amid the process ranging from component decoupling to component extrapolation. As far as we know, SCNN is the pioneer in applying a deep and **fully statistically interpretable** method in time series forecasting and achieves comparable performance to state-of-the-art models. The solution allows the practitioners to identify the information captured or disregarded by the model. In addition, it also helps to reveal the contribution of historical observations to the projection of the forthcoming observations from a statistical point of view.

Beyond modularity and interpretability, SCNN enjoys additional practical advantages, namely adaptability and scalability, over the prior works [7], [8], [9], [10], [12]. It is well-established that time series data exhibits complex distribution shifts, as shown in Fig. 2. The forms of shifts extend beyond vanilla unconditional data distribution and

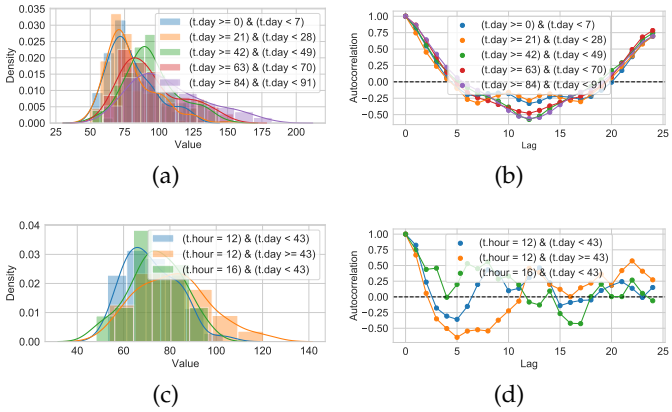


Fig. 2: Illustrations of distribution shifts of diverse forms: (a) $P(Y_t)$; (b) $Corr(Y_t, Y_{t-i})$; (c) $P(Y_t|t.hour)$; (d) $Corr(Y_t, Y_{t-i}|t.hour)$. These visualizations highlight that both the data ($P(Y_t)$) and the transitions ($Corr(Y_t, Y_{t-i})$) exhibit complex distribution shifts that are concurrently correlated with both the time span and the hour of the day.

auto-correlations, encompassing various types of conditional data distribution and auto-correlations. The generalization ability under distribution shifts remains a critical issue with the current deep learning models. We demonstrate that SCNN excels at **self-adapting to temporal shifts** in data distribution, since the structured components can capture the statistical changes in the online samples themselves. This is in contrast to the previous models, whose parameters are trained to capture the overall statistics, with each sample being treated evenly irrespective of its occurrence time. Moreover, the number of parameters involved in our model does not scale with the length of the input sequence, and the computational cost is linear to the size of the input sequence, which is more scalable than the linear regression (LR), multi-layer perceptron (MLP) or TCN-based approaches [2], [9], [10], [14], [17]. Our approach aligns with the intuition that the complexity of the underlying dynamical system determines the essential number of parameters, irrespective of the input sequence length.

We summarize our contributions as follows:

- We introduce the Structured Component Neural Network (SCNN) for multivariate time series forecasting, marking the first completely decomposition-based neural architecture.
- We propose a novel structural regularization method to explicitly shape the structure of the representation space learned from SCNN.
- We conduct extensive experiments on three public datasets to validate the effectiveness of SCNN, and observe general improvement over competing methods.
- Empirical and analytical evidence demonstrates the SCNN's superior performance in handling distribution shifts and anomalies, and maintaining computational efficiency.

2 Related Work

The time series forecasting community has undergone rapid development since the flourishing of deep learning models [18]. The vast majority of works inherit from a small group of canonical operations, consisting of the attention operator, the convolution operator and the recurrent operator. In particular, the derivatives of the attention operator include spatial attention [19], [20], [21], temporal attention [20], [22] and sparse attention (to improve computational efficiency) [11], [23], [24]; the convolution operator is developed to spatial convolution [25], [26], [27], temporal convolution [9], [10], spatial-temporal convolution [28], [29] and adaptive convolution (where the parameters of the convolution operator can adapt to external conditions) [30]; the recurrent operator stimulates the development of gated recurrent units (GRU) [31], long short-term memory (LSTM) [32], [33] and adaptive RNN [30], [34], [35], [36].

To further supplement the operations above, various tricks are created. For example, to handle cases where spatial or temporal relationships are incomplete, several studies [9], [10], [12], [37], [38], [39], [40], [41], [42], [43] make use of an adaptive graph learning module to recover the relationships from data adaptively. To incorporate domain knowledge, such as periodicity, into modeling, several studies [44], [45], [46], [47] have devised ad-hoc network architecture with handcrafted connections between neural units; another line of research [20], [48] represents knowledge with a group of learnable vectors, and feeds them into the model accompanied by MTS data. Furthermore, [49], [50] used Fourier transform to decompose original MTS data into a group of orthogonal signals; [51] resorted to memory networks to enable the long-term memory of historical observations; [52] exploited a graph ordinary differential equation (ODE) to address the over-smoothing problem of graph convolution networks; [53], [54] took advantage of neural architecture search algorithms to search for the optimal connections between different kinds of neural blocks; and [55] integrated a transformer with a state space model to provide probabilistic and interpretable forecasts.

Recently, an emerging line of approaches capitalize on the decomposition techniques to enhance the effectiveness and interpretability of time series forecasting models. [13], [16] disentangled trend and seasonal components from TS data in latent space via a series of auxiliary objectives; [15] integrated a decomposition module into the transformer framework to approach the non-stationary issue; [14], [56] proposed spatial and temporal normalization to decompose MTS data from the spatial and temporal view, respectively. The novelty of our work is that we are the first to devise a completely decomposition-based neural architecture where the components are estimated in an attentive way to allow for data-driven adaptation. Our model achieves remarkable results compared to the state-of-the-arts based on TCNs, Transformer or RNNs.

3 Preliminaries

In this section, we introduce the definitions and the assumption. All frequently used notations are reported in Table 1.

Definition 1 (Multivariate time series forecasting)**.** Multivariate time series is formally defined as a collection of random variables $\{Y_{n,t}\}_{n\in N, t\in T}$, where n denotes the index on the spatial domain and t denotes the index on the temporal domain. Time series forecasting is formulated as the following conditional distribution:

$$P(Y_{:,t+1:t+T_{\text{out}}}|Y_{:,t-T_{\text{in}}+1:t}) = \prod_{i=1}^{T_{\text{out}}} P(Y_{:,t+i}|Y_{:,t-T_{\text{in}}+1:t}).$$

TABLE 1: Notations

Notation	Description
N, L	Number of variables / network layers.
$T_{\text{in}}, T_{\text{out}}$	Number of input steps / output steps.
$Y \in \mathbb{R}^{N\times T}$	Multivariate time series.
$Y^{\text{in}}_{n,t} \in \mathbb{R}$	Observation of n^{th} variable at time t.
$\hat{Y}^{\text{out}}_{n,t+i} \in \mathbb{R}$	Mean prediction of the n^{th} variable for the i^{th} forecast horizon at time t
$\hat{\sigma}^{\text{out}}_{n,t+i} \in \mathbb{R}$	Standard deviation prediction of the n^{th} variable for the i^{th} forecast horizon. at time t.
lt, se, st, ce	Abbreviations for 4 types of structured components: long-term, seasonal, short-term, co-evolving.
$\mu^*_{n,t}, \sigma^*_{n,t} \in \mathbb{R}^{d_z}$	Historical structured component.
$\hat{\mu}^*_{n,t+i}, \hat{\sigma}^*_{n,t+i} \in \mathbb{R}^{d_z}$	Extrapolation of the structured component.
$H_{n,t} \in \mathbb{R}^{8d_z}$	Concatenation of historical structured components of 4 types.
$\hat{H}_{n,t+i} \in \mathbb{R}^{8d_z}$	Concatenation of extrapolated structured components of 4 types.
$Z^{(l)}_{n,t} \in \mathbb{R}^{d_z}$	Historical residual representation at the l^{th} layer in the decoupling block.
$\hat{Z}^{(l)}_{n,t+i} \in \mathbb{R}^{d_z}$	Extrapolation of the residual representation at the l^{th} layer.
$Z_{n,t} \in \mathbb{R}^{4d_z}$	Concatenation of historical residual representations at 4 layers.
$\hat{Z}_{n,t+i} \in \mathbb{R}^{4d_z}$	Concatenation of extrapolated residual representations at 4 layers.
$S_{n,t} \in \mathbb{R}^{d_z}$	Historical state.
$\hat{S}_{n,t+i} \in \mathbb{R}^{d_z}$	Extrapolation of the state.

Our study focuses on a typical class of time series that can be expressed as a superposition of a set of elementary signals, namely the long-term component (lt), the seasonal component (se), the short-term component (st), and the co-evolving (ce) component. These components describe the structure of the dynamic system underlying the time series from different views and can be used to enrich the information on the time series. In particular, the long-term component captures the long-term continuity; the seasonal component characterizes the seasonality; the short-term component captures the short-term continuity; and the co-evolving component captures the spatial correlations.

Definition 2 (Generative process for multivariate time series)**.** We assume that the time series is generated by the following process:

$$\begin{aligned}
Z^{(3)}_{n,t} &= \sigma^{\text{ce}}_{n,t} Z^{(4)}_{n,t} + \mu^{\text{ce}}_{n,t},\\
Z^{(2)}_{n,t} &= \sigma^{\text{st}}_{n,t} Z^{(3)}_{n,t} + \mu^{\text{st}}_{n,t},\\
Z^{(1)}_{n,t} &= \sigma^{\text{se}}_{n,t} Z^{(2)}_{n,t} + \mu^{\text{se}}_{n,t},\\
Z^{(0)}_{n,t} &= \sigma^{\text{lt}}_{n,t} Z^{(1)}_{n,t} + \mu^{\text{lt}}_{n,t},
\end{aligned}$$

where $Z_{n,t}^{(0)}$ denotes the original representation of the n^{th} TS at t, and $Z_{n,t}^{(i)}$ ($i \in \{1,2,3,4\}$) denotes the residual representation at the i^{th} level. Each structured component is represented by a scaling factor σ_t^* and a location factor μ_t^*, where $* \in \{\text{ce}, \text{st}, \text{se}, \text{lt}\}$.

4 Structured Component-based Neural Network

Figure 3 illustrates an overview of our model architecture. SCNN is composed of three major parts, namely component decoupling, component extrapolation, and structural regularization. We will introduce each part in the following sections.

4.1 Component Decoupling

This section introduces how to estimate a specific structured component, and decouple this component from the residuals by applying a normalization operator. This process is presented in the left part of Fig. 3.

4.1.1 Long-Term Component

The long-term component aims to be the characterization of the long-term patterns of the time series data. To avoid ambiguity, we refer to the pattern as the distribution of the aggregated samples without considering the chronological order among them; the long-term pattern refers to the data distribution over an extended period that should cover multiple seasons. By aggregating the samples collected from multiple seasons, we can eliminate the short-term impact that will affect only a handful of time steps, and acquire the estimation of the long-term component with less bias.

We create a sliding window of size Δ to dynamically select the set of samples over time. Then, the location (mean) and scale (standard deviation) of the samples are computed and jointly taken as the measurement of the long-term component. Finally, we transform the representation by subtracting the location from it and dividing the difference by the scale, in order to unify the long-term components for different samples. The formula takes the following form:

$$\mu_{n,t}^{\text{lt}} = \frac{1}{\Delta}\sum_{i=0}^{\Delta-1} Z_{n,t-i}^{(0)}\ , \tag{1}$$

$$(\sigma_{n,t}^{\text{lt}})^2 = \frac{1}{\Delta}\sum_{i=0}^{\Delta-1} (Z_{n,t-i}^{(0)})^2 - (\mu_{n,t}^{\text{lt}})^2 + \epsilon, \tag{2}$$

$$Z_{n,t}^{(1)} = \frac{Z_{n,t}^{(0)} - \mu_{n,t}^{\text{lt}}}{\sigma_{n,t}^{\text{lt}}}, \tag{3}$$

where $\mu_{n,t}^{\text{lt}}$ and $\sigma_{n,t}^{\text{lt}}$ are the location and the scale respectively; $Z_{n,t}^{(1)}$ is the first-layer normalized representation and passed to the following normalization layers.

Previous studies [14], [15], [57] let the ϵ to be an infinitesimal value, e.g. 0.00001, for the purpose of avoiding the division-by-zero issue. We find that this trick, however, incurs an unstable optimization process in some cases, resulting in a sub-optimal solution on the parameter space. Imagine a time series that rarely receives non-zero measurements which can be viewed as unpredictable noises. The standard deviation of this time series would be very small, leading its inverse to be exceptionally large. As a result, the noises would be undesirably magnified, driving the model to fit these chaotic patterns without any predictable structure. To alleviate this dilemma, our study sets ϵ as 1, which, on the one hand, can prevent the explosion of noises and, on the other hand, cannot dominate the original scaling factor. This simple trick is also employed by [35], but they only used it to preprocess the time series data.

4.1.2 Seasonal Component

Our study makes a mild assumption that the cycle length is invariant over time. For those applications with time-varying cycle lengths, we can resort to the Fast Fourier Transform (FFT) to automate the identification of cycle length, which is compatible with our framework and is applied in a bunch of methods like Autoformer [11].

Disentanglement of the seasonal component resembles the long-term component, except that we apply a dilated window whose dilation factor is set to the cycle length. Let τ denote the window size, and m denote the dilation factor. The normalization then proceeds as follows:

$$\mu_{n,t}^{\text{se}} = \frac{1}{\tau}\sum_{i=0}^{\tau-1} Z_{n,t-i*m}^{(1)}\ , \tag{4}$$

$$(\sigma_{n,t}^{\text{se}})^2 = \frac{1}{\tau}\sum_{i=0}^{\tau-1} (Z_{n,t-i*m}^{(1)})^2 - (\mu_{n,t}^{\text{se}})^2 + \epsilon, \tag{5}$$

$$Z_{n,t}^{(2)} = \frac{Z_{n,t}^{(1)} - \mu_{n,t}^{\text{se}}}{\sigma_{n,t}^{\text{se}}}. \tag{6}$$

In this way, the resulting $\mu_{n,t}^{\text{se}}$ and $\sigma_{n,t}^{\text{se}}$ will exhibit only seasonal patterns without interference by any temporary or short-term impacts.

4.1.3 Short-Term Component

The short-term component captures the irregular and short-term effects, which cannot be explained by either the long-term component or the seasonal component. In contrast to the long-term normalization, the window size here needs to be set to a small number, notated by δ, such that the short-term effect will not be smoothed out. Likewise, the formula takes the following form:

$$\mu_{n,t}^{\text{st}} = \frac{1}{\delta}\sum_{i=0}^{\delta-1} Z_{n,t-i}^{(2)}\ , \tag{7}$$

$$(\sigma_{n,t}^{\text{st}})^2 = \frac{1}{\delta}\sum_{i=0}^{\delta-1} (Z_{n,t-i}^{(2)})^2 - (\mu_{n,t}^{\text{st}})^2 + \epsilon, \tag{8}$$

$$Z_{n,t}^{(3)} = \frac{Z_{n,t}^{(2)} - \mu_{n,t}^{\text{st}}}{\sigma_{n,t}^{\text{st}}}. \tag{9}$$

The downside of the short-term component is that it cannot timely detect a short-term change in data, demonstrating response latency. Also, it is insensitive to changes that only endure for a limited number (e.g., two or three) of time steps. To mitigate this issue, we can make use of the contemporary measurements of the co-evolving time series.

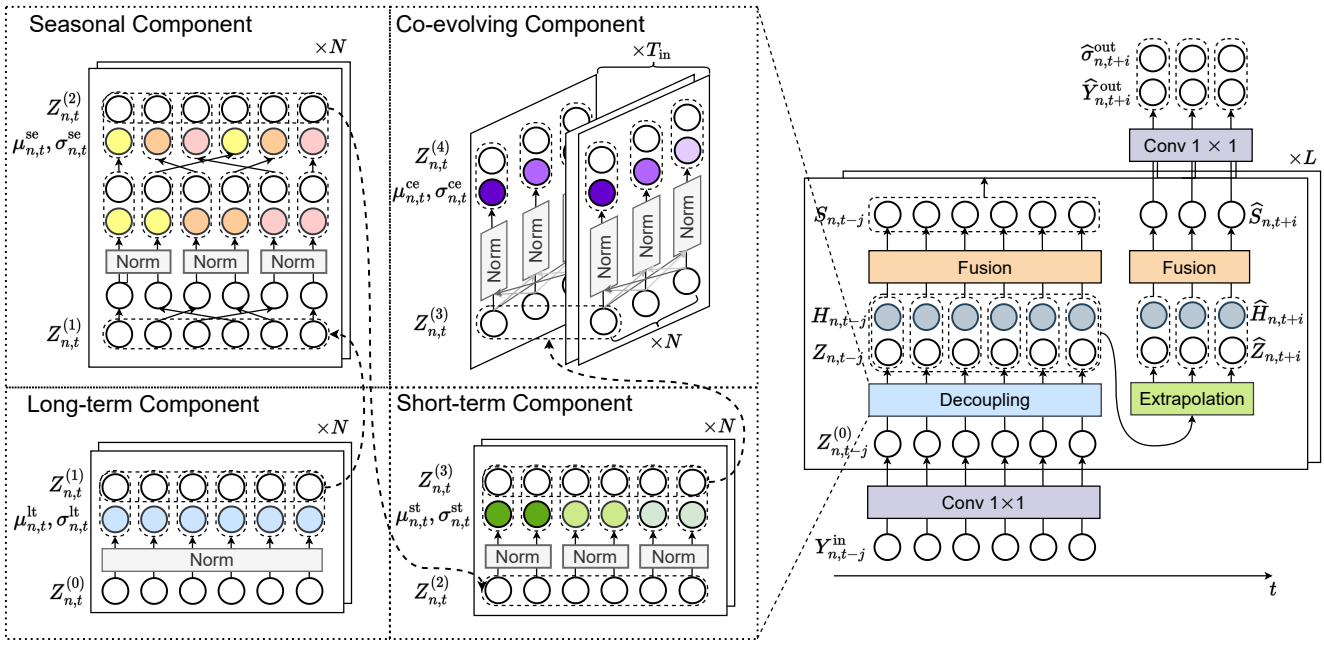


Fig. 3: A schematic diagram of SCNN.

4.1.4 Co-evolving Component

The co-evolving component, derived from the spatial correlations between time series, is advantageous for capturing instant changes in time series, which distinguishes it from the above three components. A co-evolving behavior shared across multiple time series indicates that these time series are generated from the same process. Then, we can get an estimator of this process by aggregating multiple samples drawn from it.

A key problem to be solved here is identifying which time series share the same co-evolving component. Technically, this amounts to measuring correlations between different time series. This measurement can be done either by hard-coding the correlation matrix with prior knowledge or by parameterizing and learning it. Our study adopts the latter practice, which allows for more flexibility, since many datasets do not present prior knowledge about the relationship between time series. We assign an individual attention score to every pair of time series, and then normalize the attention scores associated with the same time series via softmax to ensure that all attention scores are summed up to 1. Formally, let $\alpha_{n,n'}$ denote the attention score between the n^{th} and n'^{th} variable. The formula is written as follows:

$$a_{n,n'} = \frac{\exp(\alpha_{n,n'})}{\sum_{j=1}^{N} \exp(\alpha_{n,j})}, \tag{10}$$

$$\mu^{\text{ce}}_{n,t} = \sum_{n'=1}^{N} a_{n,n'} Z^{(3)}_{n',t}\ , \tag{11}$$

$$(\sigma^{\text{ce}}_{n,t})^2 = \sum_{n'=1}^{N} a_{n,n'} (Z^{(3)}_{n',t})^2 - (\mu^{\text{ce}}_{n,t})^2 + \epsilon, \tag{12}$$

$$Z^{(4)}_{n,t} = \frac{Z^{(3)}_{n,t} - \mu^{\text{ce}}_{n,t}}{\sigma^{\text{ce}}_{n,t}}, \tag{13}$$

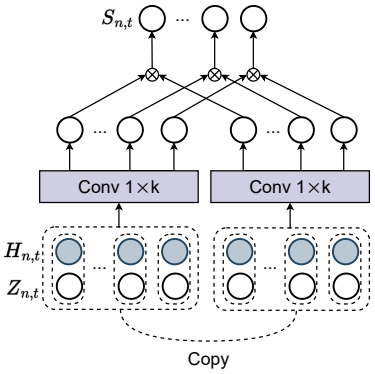


Fig. 4: Component Fusion

where $Z^{(4)}_{n,t}$ denotes the residuals that cannot be modeled by any of our proposed components. This computation can be further modified to improve the scalability via the adjacency matrix learning module proposed in [9].

The decoupled components and residual representations are sequentially concatenated to form a wide vector:

$$\begin{aligned} Z_{n,t} =& [Z^{(1)}_{n,t}\ , Z^{(2)}_{n,t}\ , Z^{(3)}_{n,t}\ , Z^{(4)}_{n,t}], \\ H_{n,t} =& [\mu^{\text{lt}}_{n,t}\ , \sigma^{\text{lt}}_{n,t}\ , \mu^{\text{se}}_{n,t}\ , \sigma^{\text{se}}_{n,t}\ , \\ & \mu^{\text{st}}_{n,t}\ , \sigma^{\text{st}}_{n,t}\ , \mu^{\text{ce}}_{n,t}\ , \sigma^{\text{ce}}_{n,t}]. \end{aligned}$$

To model interactions between the components, we couple the causal convolution operator with the element-wise multiplication operator, taking $Z_{n,t}$ as input. This computational process is graphically represented in Fig. 4, and is

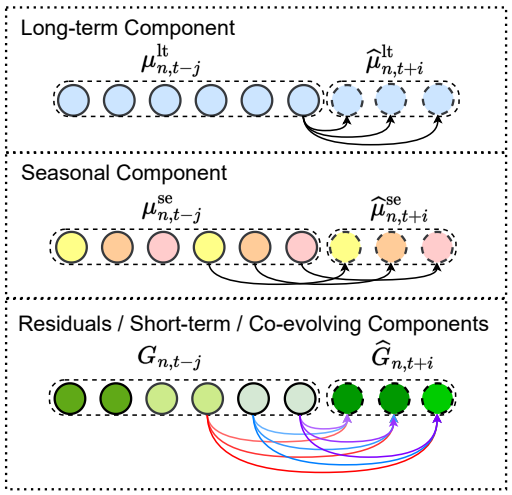


Fig. 5: Component Extrapolation

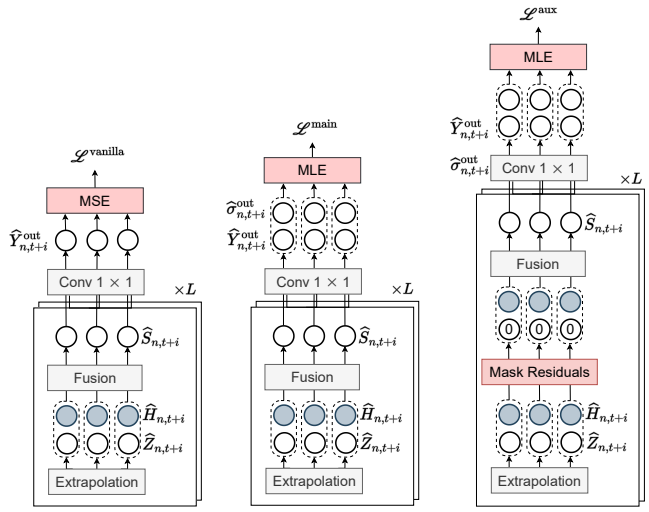


Fig. 6: Structural Regularization. The term $\mathcal{L}^{\text{vanilla}}$ denotes the standard MSE loss function. On the other hand, $\mathcal{L}^{\text{main}}$ and $\mathcal{L}^{\text{aux}}$ are specifically designed to enforce regularization within the feature space, thereby ensuring a more structured representation of the data. These two loss functions work together to optimize the model's performance.

formally written as:

$$S_{n,t} = \left(\sum_{j=0}^{k-1} W_j^{(1)}[Z_{n,t-j}, H_{n,t-j}]\right) \otimes \left(\sum_{j=0}^{k-1} W_j^{(2)}[Z_{n,t-j}, H_{n,t-j}]\right), \tag{14}$$

where k is the kernel size of the convolution operator and $W_j^{(1)}, W_j^{(2)} \in \mathbb{R}^{d_z \times 12 d_z}$ are learnable matrices. $S_{n,t}$ can be passed to another component estimation block as $Z_{n,t}^{(0)}$ to produce richer compositions of the structural components.

4.2 Component Extrapolation

We simulate the dynamics of each component with a customized and basic model. This allows for the explainability of the features being accounted for by the model and the provision of insights into the capacity of the forecasting model. With the acquired understanding of the features and the model capacity, practitioners can detect the anomaly points where the model may not present reliable results, and adopt specific measures to handle the anomalies. The components exhibit different dynamics with varying degrees of predictability, motivating us to create separate models to mimic the prospective development of their dynamics. The models are visualized in Fig. 5.

For a short period of time in the future, the long-term component and the seasonal component change in a relatively well-defined behavior, so we can directly specify the law for extrapolation without introducing extra parameters. We trivially reuse the (estimated) state of the long-term component at the current time point for the extrapolation of each future time point.

$$\hat{\mu}^{\text{lt}}_{n,t+i} = \mu^{\text{lt}}_{n,t}\ ,\ \hat{\sigma}^{\text{lt}}_{n,t+i} = \sigma^{\text{lt}}_{n,t}\ . \tag{15}$$

For the seasonal component and its residual, we also conduct replication but from the time point at the same phase as the target time point in the previous season:

$$\hat{\mu}^{\text{se}}_{n,t+i} = \mu^{\text{se}}_{n,t-m+i}\ ,\ \hat{\sigma}^{\text{se}}_{n,t+i} = \sigma^{\text{se}}_{n,t-m+i}\ . \tag{16}$$

The short-term component, the co-evolving component, and the residual representations vary with greater stochasticity and thereby less predictability than the above two components due to their irregularity. Since the dynamics are now much more complicated, we opt to parameterize the dynamical model to allow for more flexibility than specifying a fixed heuristic law. For each of these three types of representations, we employ an auto-regressive model, predicting the representation for the i^{th} forecast horizon based on the past δ representations. For the sake of brevity, we present the extrapolation processes of the short-term and co-evolving components together with the residuals in a single figure, given that they share the same model form:

$$\hat{G}_{n,t+i} = \sum_{j=0}^{\delta-1} \hat{W}_{ji} G_{n,t-j} + b_i, \tag{17}$$

where $G \in \{Z^{(l)}_{n,t+i}, \mu^{\text{st}}_{n,t+i}, \sigma^{\text{st}}_{n,t+i}, \mu^{\text{ce}}_{n,t+i}, \sigma^{\text{ce}}_{n,t+i}\}$; $\hat{W}_{ji}$, a parameter matrix of size $d_z \times d_z$, quantifies the contribution from $G_{n,t-j}$ to $\hat{G}_{n,t+i}$; b_i is the bias term. $\hat{W}_{ji}$ and b_i are subject to training.

We concatenate the extrapolated components, denoted as $\hat{H}_{n,t+i}$, and the residuals, $\hat{Z}_{n,t+i}$. We then model their interactions, parameterized by two learnable matrices, $\hat{W}^{(1)}$ and $\hat{W}^{(2)}$, both belonging to $\mathbb{R}^{d_z \times 12 d_z}$, as follows:

$$\hat{S}_{n,t+i} = \left(\hat{W}^{(1)}[\hat{Z}_{n,t+i}, \hat{H}_{n,t+i}]\right) \otimes \left(\hat{W}^{(2)}[\hat{Z}_{n,t+i}, \hat{H}_{n,t+i}]\right), \tag{18}$$

So far, we construct a projection from the past to the future, consisting of statistically meaningful operations.

4.3 Structural Regularization

Conventionally, the objective function for time series forecasting aims to minimize the mean squared errors (MSE) or

mean absolute errors (MAE) between the predictions and the ground truth observations. The assumption inherent to this objective is that all the variables share the same variance of 1. However, this does not enable the learned representations to be organized in a desired structure, where variables can see different degrees of variance at different times due to the time-varying scaling effects prescribed by the generative structure of time series. Instead, we opt to optimize the maximum likelihood estimate (MLE) [35], which allows SCNN to improve the shaping of the structure of the representation space. In addition, an auxiliary objective function is designed to improve the nuances in feature space at the component level. We graphically contrast the two designed objective functions against the vanilla MSE loss Fig. 6

We apply linear transformations to the representations output from the component extrapolation module, producing the location (i.e. mean) $\hat{Y}^{\text{out}}_{n,t+i}$ and the scale (i.e. standard deviation) $\hat{\sigma}^{\text{out}}_{n,t+i}$, where $\hat{\sigma}^{\text{out}}_{n,t+i}$ further goes through a SoftPlus function to enable itself to be non-negative. The MLE loss is written as:

$$\mathcal{L}^{\text{main}} = \sum_{n=1}^{N}\sum_{i=1}^{T_{\text{out}}}(\log(\text{SoftPlus}(\hat{\sigma}^{\text{out}}_{n,t+i})) + \frac{(Y_{n,t+i} - \hat{Y}^{\text{out}}_{n,t+i})^2}{2(\text{SoftPlus}(\hat{\sigma}^{\text{out}}_{n,t+i}))^2}).$$

The first term in the above loss function encourages the scaling factor to be small, and the second term penalizes the deviation between the extrapolated data and the ground truth data weighted by the inverse of the scaling factor.

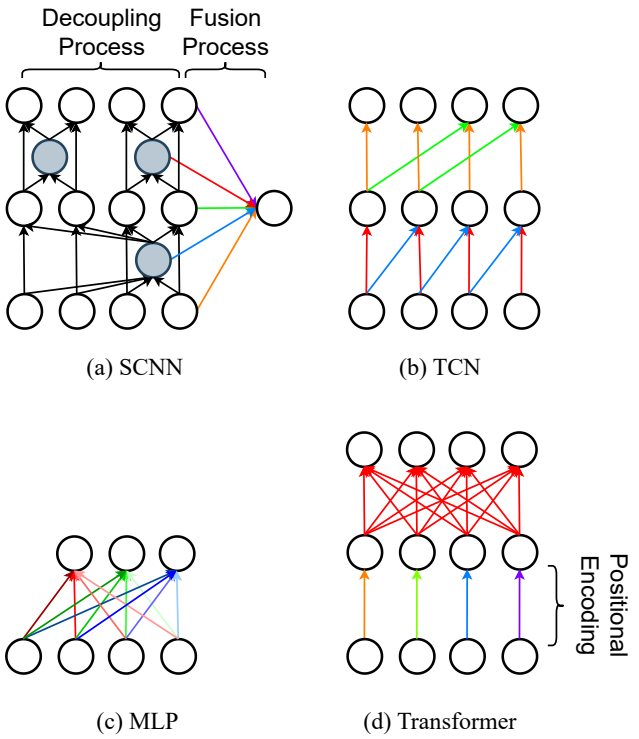
Solely leveraging the above objective to learn the forecasting dynamics does not ensure robust estimation of the structured components with their contribution to the projection. The intuition is that since the residual components, especially at the bottom levels, still contain a part of the structural information, they will take a certain amount of attributions that are supposed to belong to the structured components as learning the corresponding weights for the components. Attributing improper importance to the residual components incurs considerable degradation in the model performance, once the time series data is contaminated with random noise that heavily impacts the high-frequency signal.

To approach this issue, the basic idea is to accentuate the structured components that suffer less from corruption with an additional regularizer. This regularizer works to prompt the model to achieve a reasonable forecast using purely the structured components without the need for residual components. In particular, in the forward process of a training iteration, SCNN forks another branch after the component extrapolation module. This branch starts by zero-masking all the residual components, passing only structured components through the following operations. Finally, it yields an auxiliary pair of forecast coefficients $\hat{Y}^{\text{aux}}_{n,t+i}$ and $\hat{\sigma}^{\text{aux}}_{n,t+i}$, which are also being tailored by MLE.

The ultimate objective to be optimized is an aggregation of all the above objective functions in a weighted fashion:

$$\mathcal{L} = \alpha\mathcal{L}^{\text{aux}} + \mathcal{L}^{\text{main}}, \tag{19}$$

where α is the hyper-parameter that controls the importance of the corresponding objective. We use the Adam optimizer [58] to optimize this target.



(a) SCNN (b) TCN (c) MLP (d) Transformer

Fig. 7: Data and computational flow. Each edge symbolizes an atomic operation involving a single variable situated at the tail of the edge. If an operation is parameterized, the corresponding edge is color-coded.

4.4 Complexity Analysis

We conduct an analysis of two types of complexity associated with our model: first, the parameter complexity, which refers to the number of parameters involved in the model; and second, the computational complexity. We draw a comparison between the complexity of the SCNN and three prominent frameworks, namely the Transformer, the TCN, and the MLP.

Figure 7 provides a visual representation of the data and computational flow associated with these four frameworks. Within these diagrams, each edge symbolizes an atomic operation involving a single variable situated at the tail of the edge. If an operation is parameterized, the corresponding edge is color-coded. Edges sharing the same color denote operations utilizing the same set of learnable parameters. Within the SCNN framework, the decoupling process is carried out without parameterization, thus these edges are illustrated in black. The structured components that emerge from this process are subsequently integrated, employing component-dependent parameters.

Let's denote the number of components crafted within our model as m. The number of parameters within SCNN scales in proportion to the number of components inherent in the time series, which is $\mathcal{O}(m)$. This contrasts with the majority of state-of-the-art (SOTA) models, where the parameter count scales with the length of the input sequence. To illustrate, TCN or WaveNet-based models necessitate at least $\mathcal{O}(\log T)$ parameters to process a sequence of length T; MLP or Linear Regression (LR)-based models require $\mathcal{O}(T)$

parameters; and Transformer-based models also demand $\mathcal{O}(T)$ parameters to attain SOTA performance, as demonstrated in [59]. Our approach aligns with the principle that the complexity of the underlying dynamical system dictates the requisite number of parameters, regardless of the input sequence length.

Regarding the computational complexity relative to sequence length, SCNN attains a complexity of $\mathcal{O}(Tm)$. Notably, we can further reduce the complexity of an inference step to $\mathcal{O}(m)$ by approximating the structured component using a moving average approach. This stands in contrast to alternative methods such as the MLP, which achieves a complexity of $\mathcal{O}(Th)$, with h representing the number of units in the hidden layer. The Transformer model yields a complexity of $\mathcal{O}(T^2)$, while the TCN model reaches a complexity of $\mathcal{O}(T \log T)$. Therefore, in terms of computational complexity with respect to sequence length, the SCNN proves to be the most efficient model, particularly when the structured component is estimated in a moving average manner. This observation underscores the advantage of SCNN in scenarios where computational efficiency and scalability are critical considerations.

5 Evaluation

In this section, we conduct extensive experiments on three common datasets to validate the effectiveness of SCNN from various aspects.

5.1 Experiment Setting

5.1.1 Datasets

To evaluate the performance of our model, we conduct experiments on three real-world datasets, namely BikeNYC[1], PeMSD7[2] and Electricity[3]. The statistics and the experiment settings regarding the three datasets are reported in Table 2. We adopt the same data pre-processing strategy as most of the current works [9], [10], where the TS data of each variable is individually standardized.

TABLE 2: Dataset statistics.

Tasks	Electricity	PeMSD7	BikeNYC
Start time	10/1/2014	5/1/2012	4/1/2014
End time	12/31/2014	6/30/2012	9/30/2014
Sample rate	1 hour	30 minutes	1 hour
# Timesteps	2184	2112	4392
# Variate	336	228	128
Training size	1848	1632	3912
Validation size	168	240	240
Testing size	168	240	240
Input length	144	288	144
Output length	3	3	3

5.1.2 Network Setting

The input length is set to a multiple of the season length, so that sufficient frames governed by approximately the same seasonal and long-term components can be gathered to yield estimation without much deviation. The layer number is set to 4; The number of hidden channels d is 8; Δ is set to the same quantity as the length of the input sequence; δ is set to 8; the kernel size of the causal convolution k is configured as 2. In the training phase, the batch size is 8; the weight for the auxiliary objective α is 0.5; the learning rate of the Adam optimizer is 0.0001. We also test other configurations in the hyper-parameter analysis.

5.1.3 Evaluation Metrics

We validate our model by root mean squared error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE). We repeat the experiment ten times for each model on each dataset and report the mean of the results.

5.2 Baseline Models

We compare SCNN with the following state-of-the-art models:

- **Autoformer [11].** Autoformer is specifically designed to incorporate a decomposition mechanism within the Transformer framework.
- **LSTNet [7].** LSTNet uses CNN to extract local features and uses RNN to capture long-term dependencies. It also employs a classical auto-regressive model to address scale-insensitive limitations.
- **StemGNN [49].** StemGNN models spatial and temporal dependencies in the spectral domain.
- **GW [9].** GW proposes an adaptive graph learning module that progressively recovers the spatial correlations during training. In addition, it employs Wavenet to handle correlations in the temporal domain.
- **MTGNN [10].** MTGNN designs a graph learning module that integrates external knowledge like variable attributes to learn uni-directed relations among variables.
- **AGCRN [8].** AGCRN develops two adaptive modules to build interactions between the variables. In addition, it selects RNN to undertake the job of modeling temporal evolution.
- **SCINet [60]** SCINet proposes a downsample-convolve-interact architecture which is beneficial for integrating multi-resolution features.
- **STG-NCDE [61].** STG-NCDE takes advantage of Neural Controlled Differential Equations (NCDEs) to conduct spatial-temporal processing. It generalizes canonical RNN and CNN to continuous RNN and GCN based on NCDEs.
- **GTS [62].** GTS proposes a structure learning module to learn pairwise relationships between the variables.
- **ST-Norm [14].** ST-Norm designs two normalization modules to refine the high-frequency and local components separately from MTS data.

In order to make the comparison fair, all the competing models are fed with the same number of preceding frames as SCNN. We find that this extension of input horizons can bring performance gain to various degrees.

1. https://ride.citibikenyc.com/system-data
2. https://pems.dot.ca.gov/
3. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

TABLE 3: Performance on the BikeNYC dataset

Model	MAPE (%)			MAE			RMSE		
	Horizon 1	Horizon 2	Horizon 3	Horizon 1	Horizon 2	Horizon 3	Horizon 1	Horizon 2	Horizon 3
Autoformer	20.1	21.1	22.8	3.01	3.11	3.38	6.17	6.44	7.08
LSTNet	21.2	22.3	23.8	2.71	2.91	3.15	5.80	6.34	6.97
StemGNN	19.0	20.8	22.5	2.50	2.74	2.93	5.25	6.09	6.62
AGCRN	17.4	18.8	20.5	2.28	2.50	2.68	4.74	5.50	5.97
GW	18.2	19.5	20.9	2.35	2.57	2.75	4.83	5.56	6.06
MTGNN	18.0	19.5	20.9	2.35	2.57	2.73	4.87	5.69	6.18
SCINet	17.9	19.8	21.4	2.38	2.68	2.94	4.88	5.78	6.60
STG-NCDE	18.7	20.6	22.2	2.40	2.67	2.90	5.04	5.86	6.56
GTS	20.6	23.6	26.7	2.38	2.58	2.74	4.85	5.53	6.01
ST-Norm	17.3	18.6	19.9	2.26	2.46	2.62	4.66	5.38	5.84
SCNN	**16.5**	**17.3**	**18.4**	**2.13**	**2.27**	**2.40**	**4.44**	**5.02**	**5.42**
Imp	+4.6%	+6.9%	+7.5%	+5.7%	+7.7%	+8.3%	+4.7%	+6.6%	+7.1%

TABLE 4: Performance on the PeMSD7 dataset

Model	MAPE (%)			MAE			RMSE		
	Horizon 1	Horizon 2	Horizon 3	Horizon 1	Horizon 2	Horizon 3	Horizon 1	Horizon 2	Horizon 3
Autoformer	9.01	9.41	9.86	4.57	4.75	5.03	6.85	7.15	7.38
LSTNet	7.48	7.77	8.19	3.58	3.71	3.90	6.24	6.40	6.64
StemGNN	5.50	7.33	8.09	2.65	3.49	3.84	4.55	5.99	6.53
AGCRN	4.97	6.49	7.21	2.35	3.02	3.34	4.29	5.57	6.10
GW	5.02	6.56	7.10	2.39	3.10	3.35	4.28	5.51	5.94
MTGNN	5.32	6.71	7.31	2.57	3.15	3.44	4.36	5.56	6.01
SCINet	5.16	6.72	7.23	2.47	3.18	3.45	4.31	5.60	6.05
STG-NCDE	4.94	6.63	7.58	2.32	3.06	3.47	4.42	5.91	6.70
GTS	5.35	6.97	7.70	2.53	3.26	3.58	4.42	5.74	6.30
ST-Norm	4.76	6.27	7.03	2.27	2.98	3.36	4.21	5.54	6.07
SCNN	**4.47**	**5.92**	**6.50**	**2.10**	**2.75**	**2.99**	**4.06**	**5.29**	**5.76**
Imp	+6%	+5.5%	+7.5%	+7.4%	+7.7%	+10%	+3.5%	+3.9%	+3.7%

TABLE 5: Performance on the Electricity dataset

Model	MAPE (%)			MAE			RMSE		
	Horizon 1	Horizon 2	Horizon 3	Horizon 1	Horizon 2	Horizon 3	Horizon 1	Horizon 2	Horizon 3
Autoformer	22.1	22.1	21.9	32.5	32.4	32.5	67.0	68.0	68.7
LSTNet	22.4	23.0	24.8	31.1	31.8	33.8	61.2	62.6	66.8
StemGNN	10.8	13.7	15.7	15.5	19.6	22.3	34.3	43.9	49.7
AGCRN	11.4	15.6	18.0	17.3	23.0	26.4	38.9	51.2	57.9
GW	11.3	15.6	17.3	16.3	22.0	24.3	32.5	43.6	48.7
MTGNN	10.2	13.9	16.0	14.4	19.4	22.2	29.8	40.3	46.5
SCINet	10.3	13.7	16.2	14.7	20.2	23.6	33.2	44.0	51.7
STG-NCDE	10.9	14.2	16.0	16.2	21.1	23.7	36.3	47.7	52.9
GTS	10.0	14.2	17.1	14.1	19.0	22.1	31.6	42.5	48.2
ST-Norm	10.2	13.2	15.3	15.2	19.8	22.8	32.3	42.9	50.2
SCNN	**7.69**	**10.5**	**12.2**	**11.1**	**15.0**	**17.3**	**23.9**	**32.9**	**38.4**
Imp	+23.1%	+20.4%	+20.2%	21.9%	+20.9%	+21.7%	+19.7%	+18.3%	+17.4%

5.3 Experiment Results

The experiment results on the three datasets are respectively reported in Table 3, Table 4, and Table 5. It is evident that the performance of SCNN surpasses that of the baseline models by 4% to 20%, especially when performing forecasts for multi-step ahead. This is because SCNN can extract the structured components with a well-conditioned deviation. As we know, raw data contains much noise, unavoidably interfering with the quality of the extracted components. SCNN can effectively deal with this issue according to the central limit theorem. In contrast, all the benchmark models, except ST-Norm, did not explicitly account for the structured components. For example, SCINet, one of the most up-to-date state-of-the-art models, struggled to achieve competitive performance in short-term MTS forecasting, due to its deficiency in adapting to the short-term distribution shift even with the enhancement of RevIN module proposed by [57]. GTS, GW, MTGNN and AGCRN were capable of learning the spatial correlations across the variables to estimate the translating effect of a co-evolving component, but were insusceptible to the changes in its scaling effects over time. ST-Norm could decouple the long-term component and the global component (a reduced form of co-evolving component), but did not introduce the constraint to the structure of feature space.

5.3.1 Adaptability

The data patterns for the first and last few days covered by the datasets are compared in Fig. 8. The solid line denotes the seasonal mean of MTS; the bind denotes the evolution

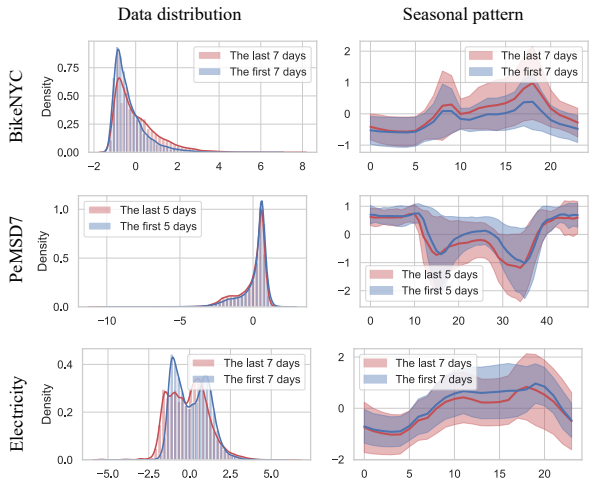


Fig. 8: Changes in data patterns as time evolves.

TABLE 6: Ablation Study

Models	BikeNYC	PeMSD7	Electricity
w/o μ^{lt} and σ^{lt}	5.12	5.04	32.7
w/o μ^{se} and σ^{se}	5.35	5.37	37.8
w/o μ^{st} and σ^{st}	5.11	5.08	33.7
w/o μ^{ce} and σ^{ce}	5.56	5.17	32.5
w/o scaling	4.98	5.05	35.6
w/o non-negligible ϵ	5.50	5.12	**30.6**
vanilla MSE loss	5.22	5.10	32.1
SCNN	**4.96**	**5.03**	31.0

of the interval between (mean - std, mean + std). It is worth noting that the data patterns for the three datasets, especially the Electricity dataset, show systematic changes from the beginning to the end. As SCNN captures the data patterns on the fly, it can automatically adapt to these statistical changes, which explains that the performance of SCNN, especially when evaluated on Electricity, exceeds that of the other competing methods by a wide margin.

5.4 Ablation Study

We design several variants, each of which is without a specific ingredient to be validated. We evaluate these variants on all three datasets and report the overall results on RMSE in Table 6. It is evident that each component can contribute to the performance of the model, but to different degrees across the three datasets. The co-evolving component is ranked as the most advantageous component in the BikeNYC task. This is because the co-evolving component incorporates the spectrum of effects ranging from long-term to short-term, and can be estimated with reasonable accuracy when the number of co-evolving variables is adequately large, which is the case for the BikeNYC data. The modeling of the long-term component only brings incremental gain to the PeMSD7 task since the training data and the testing data share an identical distribution. The scaling transformation results in significant improvement in the Electricity dataset, owing to its unification of the variables showing great differences in variance. The non-negligible ϵ, as introduced in the last paragraph of Sec. 4.1.1, is particularly useful for training SCNN on the BikeNYC dataset, as a part of TS in this dataset is very scarce, having only a handful of irregular non-zero measurements. In contrast to the vanilla MSE loss, the structural regularization can shape the structure of the feature space, preventing the overfitting issue and unlocking more power from the structured components.

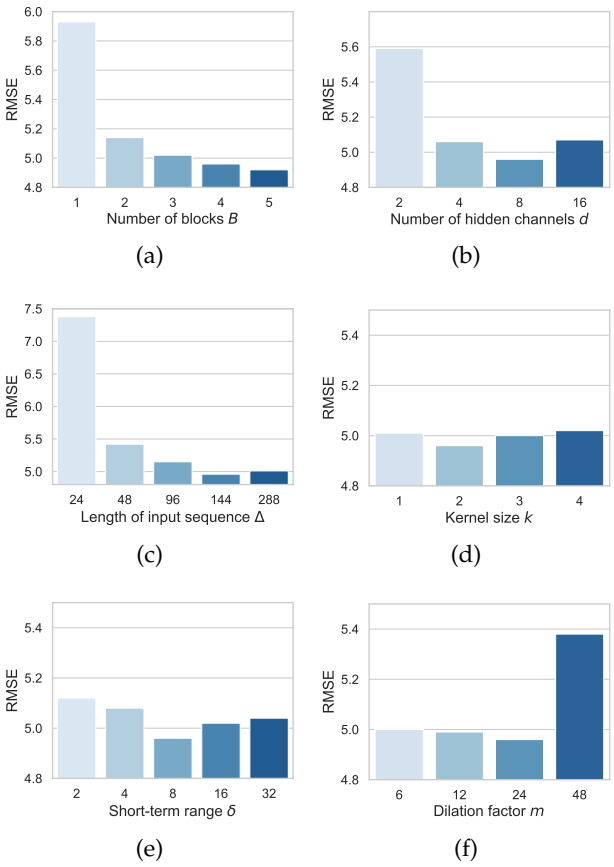


Fig. 9: Hyper-parameter analysis on BikeNYC data.

5.5 Hyper-Parameter Analysis

As shown in Fig. 9a, it is surprising that a 2-layer SCNN achieves fairly good performance, and more layers only result in incremental improvements. This demonstrates that shallow layers work on coarse-grained prediction, and deep layers perform fine-grained calibration by capturing the detailed changes presented in the MTS data. Fig. 9b shows

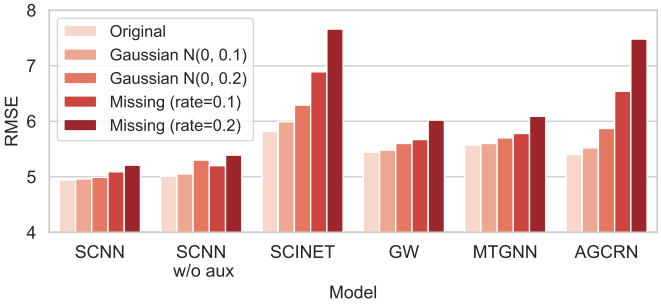
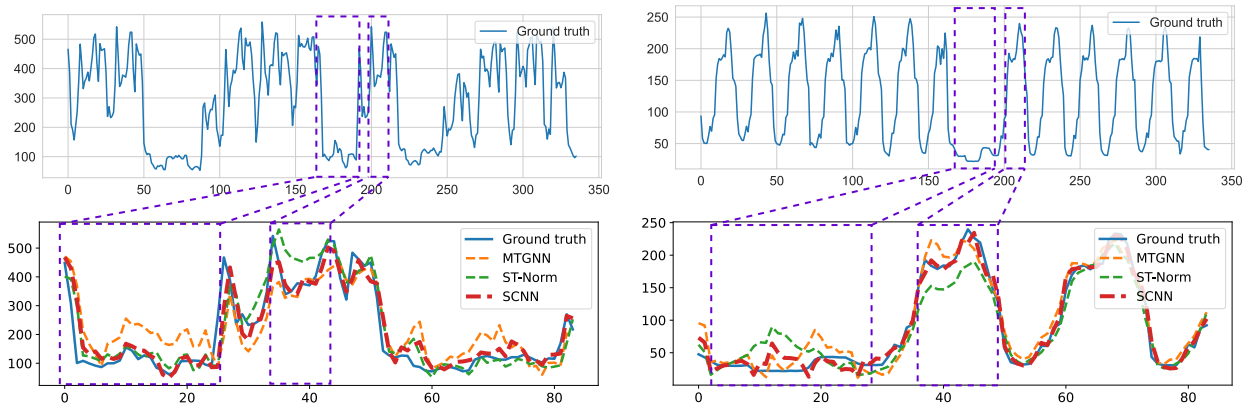
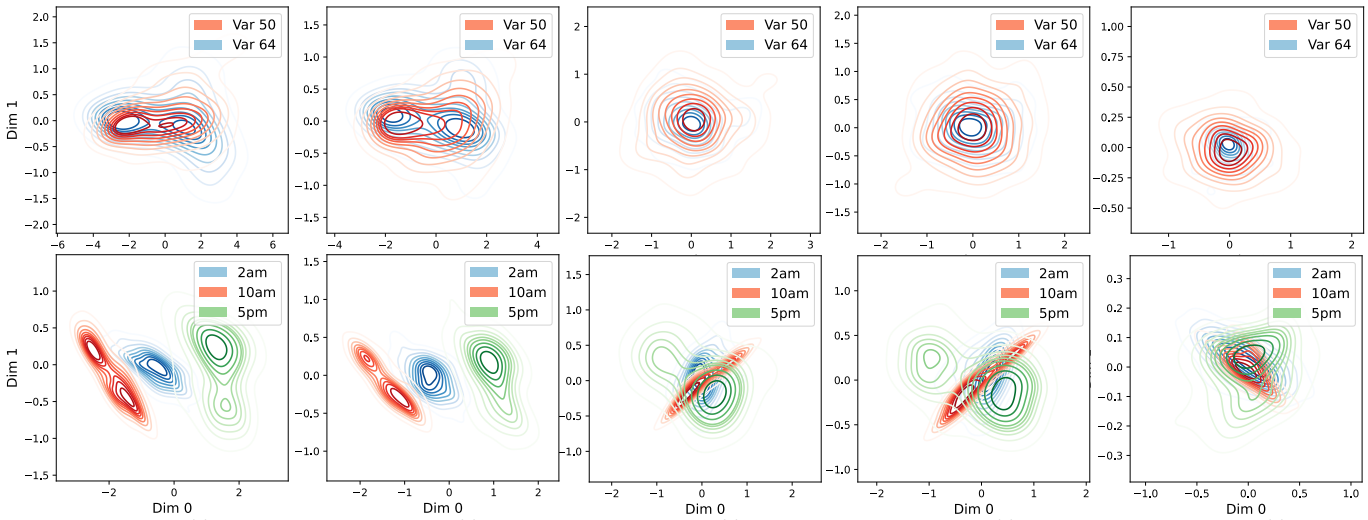


Fig. 10: Comparison of robustness.



(a) (b)

Fig. 11: Case Studies. The results demonstrate that the SCNN consistently achieves the lowest prediction error among the three models in diverse and challenging scenarios of distribution shifts and anomalies.



(a) $Z_{n,t}^{(0)}$ (b) $Z_{n,t}^{(1)}$ (c) $Z_{n,t}^{(2)}$ (d) $Z_{n,t}^{(3)}$ (d) $Z_{n,t}^{(4)}$

Fig. 12: Visualization of residual representations.

that the prediction error of SCNN firstly decreases and then increases as the number of hidden channels increases. The number of input steps can affect the estimation of the long-term component and the seasonal component, thereby leading to differences in the accuracy of the forecast, as illustrated in Fig. 9c. It is appealing to find from Fig. 9d that SCNN behaves competitively with the kernel of size 1, which means that the correlations across the local observations vanish once conditioned on the set of structured components. Fig. 9e and Fig. 9f demonstrate the effectiveness of the setup of the other two hyper-parameters.

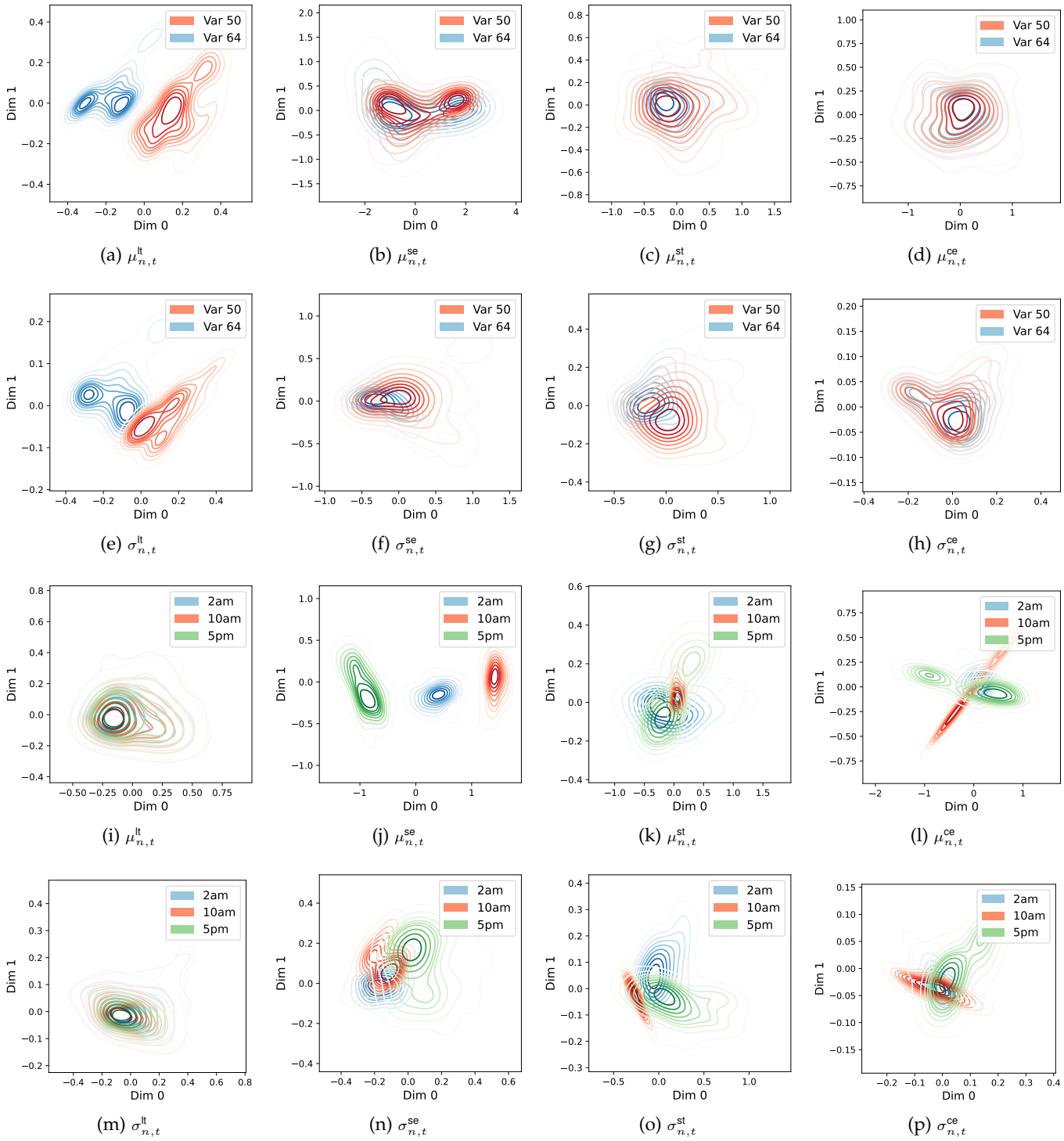
5.6 Robustness

To evaluate model robustness, we subject each model to two commonly encountered data corruptions: i.i.d. Gaussian noise and missing data. The less a model's performance degrades in the presence of these corruptions, the more robust it can be considered. In our comparison, we include SCNN, SCNN w/o aux, SCINET, GW, MTGNN, and AGCRN, with 'SCNN w/o aux' denoting the SCNN model without the structural regularization module enabled.

As demonstrated in Fig. 10, SCNN consistently exhibits the smallest performance degradation among all models under each type of corruption. This is true even when compared to SCNN w/o aux, which underlines the important role of the structural regularization module in enhancing SCNN's robustness. These results underscore SCNN's superior robustness relative to the other models examined, highlighting its resilience in the face of data corruption.

5.7 Case Studies

We provide evidence through two case studies that the SCNN consistently outperforms two competitive baselines, MTGNN and ST-Norm, particularly when dealing with anomalous patterns. This is illustrated in Fig. 11. The left figure represents an episode of a time series demonstrating irregular behavior, while the right figure exhibits another



(a) $\mu^{\text{lt}}_{n,t}$ (b) $\mu^{\text{se}}_{n,t}$ (c) $\mu^{\text{st}}_{n,t}$ (d) $\mu^{\text{ce}}_{n,t}$

(e) $\sigma^{\text{lt}}_{n,t}$ (f) $\sigma^{\text{se}}_{n,t}$ (g) $\sigma^{\text{st}}_{n,t}$ (h) $\sigma^{\text{ce}}_{n,t}$

(i) $\mu^{\text{lt}}_{n,t}$ (j) $\mu^{\text{se}}_{n,t}$ (k) $\mu^{\text{st}}_{n,t}$ (l) $\mu^{\text{ce}}_{n,t}$

(m) $\sigma^{\text{lt}}_{n,t}$ (n) $\sigma^{\text{se}}_{n,t}$ (o) $\sigma^{\text{st}}_{n,t}$ (p) $\sigma^{\text{ce}}_{n,t}$

Fig. 13: Visualization of structured components.

episode characterized by a distinct and primarily regular daily cycle.

In examining both regular and irregular episodes, we focus on two specific periods and plot the rolling predictions—predictions made on a rolling basis using a sliding window of data—for the initial forecast horizon as generated by the three models during these periods. The results demonstrate that the SCNN consistently achieves the lowest prediction error among the three models in all four scenarios. This indicates the efficacy of our design in enabling the SCNN to effectively handle anomalies or distribution shifts in a variety of contexts. These results underscore the potential of SCNN to deliver reliable and robust forecasting in diverse and challenging scenarios.

5.8 Qualitative Study

We conduct a qualitative study to cast light on how the structure of representation space is progressively reshaped

by iteratively disentangling the structured components. The structured components are visualized in Fig. 13. For the sake of visualization, we apply principal component analysis (PCA) to obtain the two-dimensional embeddings of the residual representations. Then, to convey the characteristics of the structure for any component, we perform two coloring schemes, where the first scheme, as shown in the first row of Fig. 12, separates the data points according to their spatial identities, and the second one, displayed in the second row of Fig. 12, respects their temporal identities. For clarity, we plot the kernel density estimate (KDE) for each group of points. It is conspicuous that by progressively removing the structured components from $Z_{n,t}^{(0)}$, the residual representations with different spatial and temporal identities gradually align together, suggesting that the distinct structural information has been held by the structured components.

6 Conclusion and Future Work

In this study, we put forth a generative perspective for multivariate time-series (MTS) data and accordingly present the Structured Component Neural Network (SCNN). Comprising modules for component decoupling, extrapolation, and structural regularization, the SCNN refines a variety of structured components from MTS data. Our experimental results affirm the efficacy and efficiency of the SCNN. We also conduct a series of case studies, ablation studies, and hyper-parameter analyses to perform in-depth analyses on SCNN. The model's robustness is tested against common data corruptions, such as Gaussian noise and missing data, and it consistently exhibits the smallest performance degradation among all models under each type of corruption. Furthermore, SCNN is shown to be highly effective in handling diverse and challenging scenarios, including distribution shifts and anomalies, and exhibits superior robustness compared to other models.

Looking forward, our future research will explore the potential for automating the process of identifying the optimal neural architecture, using these fundamental modules and operations as building blocks. This approach promises to alleviate the laborious task of manually testing various combinations in search of the optimal architecture for each new dataset encountered. Moreover, we anticipate that this strategy could aid in uncovering the structures and meta-knowledge inherent in time-series data. For instance, time series with complex dynamics may require high-order interactions among the structured and residual components, necessitating a large-scale neural network comprising numerous modules and complex interconnections. Extending this line of inquiry, we could discern commonalities and differences between various datasets based on the neural architectures trained on them. This represents an exciting direction for future work, potentially unveiling deeper insights into time-series analysis.

7 Acknowledgments

This work was supported in part by ARC under Grants DP180100106 and DP200101328.

References

[1] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-beats: Neural basis expansion analysis for interpretable time series forecasting," in *International Conference on Learning Representations*, 2019.

[2] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[3] R. Jiang, X. Song, D. Huang, X. Song, T. Xia, Z. Cai, Z. Wang, K.-S. Kim, and R. Shibasaki, "Deepurbanevent: A system for predicting citywide crowd dynamics at big events," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 2114–2122.

[4] R. Lam, A. Sanchez-Gonzalez, M. Willson, P. Wirnsberger, M. Fortunato, A. Pritzel, S. Ravuri, T. Ewalds, F. Alet, Z. Eaton-Rosen *et al.*, "Graphcast: Learning skillful medium-range global weather forecasting," *arXiv preprint arXiv:2212.12794*, 2022.

[5] L. Li, M. Pagnucco, and Y. Song, "Graph-based spatial transformer with memory replay for multi-future pedestrian trajectory prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 2231–2241.

[6] R. H. Shumway, D. S. Stoffer, and D. S. Stoffer, *Time series analysis and its applications*. Springer, 2000, vol. 3.

[7] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 95–104.

[8] L. BAI, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[9] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *International Joint Conference on Artificial Intelligence 2019*. Association for the Advancement of Artificial Intelligence (AAAI), 2019, pp. 1907–1913.

[10] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 753–763.

[11] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[12] X. Zhang, C. Huang, Y. Xu, L. Xia, P. Dai, L. Bo, J. Zhang, and Y. Zheng, "Traffic flow forecasting with spatial-temporal graph diffusion network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 17, 2021, pp. 15 008–15 015.

[13] Z. Wang, X. Xu, G. Trajcevski, W. Zhang, T. Zhong, and F. Zhou, "Learning latent seasonal-trend representations for time series forecasting," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022. [Online]. Available: https://openreview.net/forum?id=C9yUwd72yy

[14] J. Deng, X. Chen, R. Jiang, X. Song, and I. W. Tsang, "St-norm: Spatial and temporal normalization for multi-variate time series forecasting," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 269–278.

[15] Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Rethinking the stationarity in time series forecasting," *arXiv preprint arXiv:2205.14415*, 2022.

[16] G. Woo, C. Liu, D. Sahoo, A. Kumar, and S. Hoi, "Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting," in *International Conference on Learning Representations*, 2021.

[17] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" 2023.

[18] R. Jiang, D. Yin, Z. Wang, Y. Wang, J. Deng, H. Liu, Z. Cai, J. Deng, X. Song, and R. Shibasaki, "Dl-traff: Survey and benchmark of deep learning models for urban traffic prediction," in *Proceedings of the 30th ACM international conference on information & knowledge management*, 2021, pp. 4515–4525.

[19] S. Fang, Q. Zhang, G. Meng, S. Xiang, and C. Pan, "Gstnet: Global spatial-temporal network for traffic flow prediction." in *IJCAI*, 2019, pp. 2286–2293.

[20] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 1234–1241.

[21] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "Geoman: Multi-level attention networks for geo-sensory time series prediction." in *IJCAI*, vol. 2018, 2018, pp. 3428–3434.

[22] X. Zhou, Y. Shen, Y. Zhu, and L. Huang, "Predicting multi-step citywide passenger demands using attention-based neural networks," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 736–744.

[23] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of AAAI*, 2021.

[24] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," *Advances in Neural Information Processing Systems*, vol. 32, pp. 5243–5253, 2019.

[25] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 3634–3640.

[26] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations*, 2018.

[27] K. Guo, Y. Hu, Y. Sun, S. Qian, J. Gao, and B. Yin, "Hierarchical graph convolution networks for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, 2021, pp. 151–159.

[28] S. Guo, Y. Lin, S. Li, Z. Chen, and H. Wan, "Deep spatial–temporal 3d convolutional neural networks for traffic data forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3913–3926, 2019.

[29] S. Yang, J. Liu, and K. Zhao, "Space meets time: Local spacetime neural network for traffic flow forecasting," in *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021, pp. 817–826.

[30] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1720–1730.

[31] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[32] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 5668–5675.

[33] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[34] Y. Wang, A. Smola, D. Maddix, J. Gasthaus, D. Foster, and T. Januschowski, "Deep factors for forecasting," in *International conference on machine learning*. PMLR, 2019, pp. 6607–6617.

[35] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "Deepar: Probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.

[36] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, "Deep state space models for time series forecasting," *Advances in neural information processing systems*, vol. 31, pp. 7785–7794, 2018.

[37] M. Li and Z. Zhu, "Spatial-temporal fusion graph neural networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 5, 2021, pp. 4189–4196.

[38] L. Han, B. Du, L. Sun, Y. Fu, Y. Lv, and H. Xiong, "Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 547–555.

[39] Y. Liu, Q. Liu, J.-W. Zhang, H. Feng, Z. Wang, Z. Zhou, and W. Chen, "Multivariate time-series forecasting with temporal polynomial graph neural networks," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022. [Online]. Available: https://openreview.net/forum?id=pMumil2EJh

[40] S. Lan, Y. Ma, W. Huang, W. Wang, H. Yang, and P. Li, "DSTAGNN: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting," in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 11 906–11 917. [Online]. Available: https://proceedings.mlr.press/v162/lan22a.html

[41] J. Ye, Z. Liu, B. Du, L. Sun, W. Li, Y. Fu, and H. Xiong, "Learning the evolutionary and multi-scale graph structure for multivariate time series forecasting," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 2296–2306. [Online]. Available: https://doi.org/10.1145/3534678.3539274

[42] Z. Shao, Z. Zhang, F. Wang, and Y. Xu, "Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 1567–1577. [Online]. Available: https://doi.org/10.1145/3534678.3539396

[43] R. Jiang, Z. Wang, J. Yong, P. Jeph, Q. Chen, Y. Kobayashi, X. Song, S. Fukushima, and T. Suzumura, "Spatio-temporal meta-graph learning for traffic forecasting," *arXiv preprint arXiv:2211.14701*, 2022.

[44] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 3656–3663.

[45] D. Chai, L. Wang, and Q. Yang, "Bike flow prediction with multi-graph convolutional networks," in *Proceedings of the 26th ACM SIGSPATIAL international conference on advances in geographic information systems*, 2018, pp. 397–400.

[46] A. Zonoozi, J.-j. Kim, X.-L. Li, and G. Cong, "Periodic-crn: A convolutional recurrent model for crowd density prediction with recurring periodic patterns." in *IJCAI*, 2018, pp. 3732–3738.

[47] C. Chen, K. Li, S. G. Teo, X. Zou, K. Wang, J. Wang, and Z. Zeng, "Gated residual recurrent graph neural networks for traffic prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 485–492.

[48] J. Deng, X. Chen, Z. Fan, R. Jiang, X. Song, and I. W. Tsang, "The pulse of urban transport: exploring the co-evolving pattern for spatio-temporal forecasting," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 15, no. 6, pp. 1–25, 2021.

[49] D. Cao, Y. Wang, J. Duan, C. Zhang, X. Zhu, C. Huang, Y. Tong, B. Xu, Y. Bai, J. Tong *et al.*, "Spectral temporal graph neural network for multivariate time-series forecasting," *Proceedings of the NeurIPS 2020*, 2020.

[50] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 27 268–27 286.

[51] H. Yao, Y. Liu, Y. Wei, X. Tang, and Z. Li, "Learning from multiple cities: A meta-learning approach for spatial-temporal prediction," in *The World Wide Web Conference*, 2019, pp. 2181–2191.

[52] Z. Fang, Q. Long, G. Song, and K. Xie, "Spatial-temporal graph ode networks for traffic flow forecasting," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 364–373.

[53] Z. Pan, S. Ke, X. Yang, Y. Liang, Y. Yu, J. Zhang, and Y. Zheng, "Autostg: Neural architecture search for predictions of spatio-temporal graph," in *Proceedings of the Web Conference 2021*, 2021, pp. 1846–1855.

[54] T. Li, J. Zhang, K. Bao, Y. Liang, Y. Li, and Y. Zheng, "Autost: Efficient neural architecture search for spatio-temporal prediction," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 794–802.

[55] Y. Lin, I. Koprinska, and M. Rana, "Ssdnet: State space decomposition neural network for time series forecasting," in *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021, pp. 370–378.

[56] J. Deng, X. Chen, R. Jiang, X. Song, and I. W. Tsang, "A multi-view multi-task learning framework for multi-variate time series forecasting," 2021.

[57] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in *International Conference on Learning Representations*, 2021.
[58] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[59] Y. Zhang and J. Yan, "Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting," in *The Eleventh International Conference on Learning Representations*, 2023.
[60] M. Liu, A. Zeng, M. Chen, Z. Xu, Q. Lai, L. Ma, and Q. Xu, "Scinet: Time series modeling and forecasting with sample convolution and interaction," *Thirty-sixth Conference on Neural Information Processing Systems (NeurIPS), 2022*, 2022.
[61] J. Choi, H. Choi, J. Hwang, and N. Park, "Graph neural controlled differential equations for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.
[62] C. Shang, J. Chen, and J. Bi, "Discrete graph structure learning for forecasting multiple time series," in *International Conference on Learning Representations*, 2021.

Jinliang Deng received a B.S. degree in computer science from Peking University in 2017, and a M.S. degree in computer science from The Hong Kong University of Science and Technology in 2019. He is currently a Ph.D. candidate in the Australian Artificial Intelligence Institute, University of Technology Sydney and the Department of Computer Science and Engineering, Southern University of Science and Technology. His research interests include time series forecasting, urban computing and deep learning.

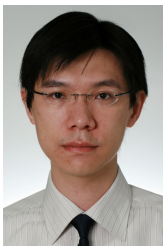
Xiusi Chen received a B.S. degree and a M.S. degree in computer science from Peking University, in 2015 and 2018, respectively. He is currently a Ph.D. candidate in the Department of Computer Science, University of California, Los Angeles. His research interests include natural language processing, knowledge graph, neural maching reasoning and reinforcement learning.

Renhe Jiang received a B.S. degree in software engineering from the Dalian University of Technology, China, in 2012, a M.S. degree in information science from Nagoya University, Japan, in 2015, and a Ph.D. degree in civil engineering from The University of Tokyo, Japan, in 2019. From 2019, he has been an Assistant Professor at the Information Technology Center, The University of Tokyo. His research interests include ubiquitous computing, deep learning, and spatio-temporal data analysis.

Du Yin received his B.S. degree in Electronic Information School of Wuhan University, Wuhan, China, in 2017, M.S. degree in the Department of Computer Science and Engineering from Southern University of Science and Technology, Shenzhen, China, in 2022. He is currently persuing a Ph.D. degree with the Department of Computer Science and Engineering, UNSW, Sydney. His research interests include deep learning, spatio-temporal traffic data mining, urban computing and big data.

Yi Yang received a B.S. degree in computer science from Beijing University of Post and communication in 2017, and a M.S. degree in computer science from The Hong Kong University of Science and Technology in 2018. He is currently a DataScience engineer Tencent Technology. His research interests include time series forecasting,casucal inferece, computing and deep learning.

Prof. Xuan Song received a Ph.D. degree in signal and information processing from Peking University in 2010. In 2017, he was selected as Excellent Young Researcher of Japan MEXT. He has served as Associate Editor, Guest Editor, Area Chair, Senior Program Committee Member for many prestigious journals and top-tier conferences, such as IMWUT, IEEE Transactions on Multimedia, WWW Journal, ACM TIST, IEEE TKDE, Big Data Journal, UbiComp, IJCAI, AAAI, ICCV, CVPR etc. His main research interests are AI and its related research areas, such as data mining and urban computing. To date, he has published more than 100 technical publications in journals, book chapters, and international conference proceedings, including more than 60 high-impact papers in top-tier publications for computer science. His research has been featured in many Chinese, Japanese and international venues, including the United Nations, the Discovery Channel, and Fast Company Magazine. He received the Honorable Mention Award at UbiComp 2015.



Ivor W. Tsang is the Director of A*STAR Centre for Frontier AI Research. He is a Professor of artificial intelligence with the University of Technology Sydney, Ultimo, NSW, Australia, and the Research Director of the Australian Artificial Intelligence Institute. His research interests include transfer learning, deep generative models, learning with weakly supervision, Big Data analytics for data with extremely high dimensions in features, samples and labels. In 2013, he was the recipient of the ARC Future Fellowship for his outstanding research on Big Data analytics and large-scale machine learning. In 2019, his JMLR paper Towards ultrahigh dimensional feature selection for Big Data was the recipient of the International Consortium of Chinese Mathematicians Best Paper Award. In 2020, he was recognized as the AI 2000 AAAI/IJCAI Most Influential Scholar in Australia for his outstanding contributions to the field between 2009 and 2019. His research on transfer learning granted him the Best Student Paper Award at CVPR 2010 and the 2014 IEEE TMM Prize Paper Award. Recently, he was conferred the IEEE Fellow for his outstanding contributions to large-scale machine learning and transfer learning. He serves as the Editorial Board for the JMLR, MLJ, JAIR, IEEE TPAMI, IEEE TAI, IEEE TBD, and IEEE TETCI. He serves/served as a AC or Senior AC for NeurIPS, ICML, AAAI and IJCAI, and the steering committee of ACML.