Joint Hypergraph Rewiring and Memory-Augmented Forecasting Techniques in Digital Twin Technology

Sagar Srinivas **Sakhinana**[*,†], Shivam **Gupta**[‡], Krishna Sai Sudhir **Aripirala**[‡] and Venkataramana Runkana

TCS Research

Abstract

Digital Twin technology creates virtual replicas of physical objects, processes, or systems by replicating their properties, data, and behaviors. This advanced technology offers a range of intelligent functionalities, such as modeling, simulation, and data-driven decision-making, that facilitate design optimization, performance estimation, and monitoring operations. Forecasting plays a pivotal role in Digital Twin technology, as it enables the prediction of future outcomes, supports informed decision-making, minimizes risks, driving improvements in efficiency, productivity, and cost reduction. Recently, Digital Twin technology has leveraged Graph forecasting techniques in large-scale complex sensor networks to enable accurate forecasting and simulation of diverse scenarios, fostering proactive and data-driven decision-making. However, existing Graph forecasting techniques lack scalability for many real-world applications. They have limited ability to adapt to non-stationary environments, retain past knowledge, lack a mechanism to capture the higher-order spatio-temporal dynamics, and estimate uncertainty in model predictions. To surmount the challenges, we introduce a hybrid architecture that enhances the hypergraph representation learning backbone by incorporating fast adaptation to new patterns and memory-based retrieval of past knowledge. This balance aims to improve the slowly-learned backbone and achieve better performance in adapting to recent changes. In addition, it models the time-varying uncertainty of multi-horizon forecasts, providing estimates of prediction uncertainty. Our forecasting architecture has been validated through ablation studies and has demonstrated promising results across multiple benchmark datasets, surpassing state-of-the-art forecasting methods by a significant margin.

Keywords

Digital Twins, Deep Learning on Graphs, Time-series Forecasting

1. Introduction

Digital twins have several applications in various domains, including finance, retail and e-commerce, logistics and transport, healthcare, and many other domains. Digital Twins are useful in finance for risk management, trading, and investment decision-making. They enable financial institutions to simulate different scenarios and identify potential risks before they occur. They can help traders identify profitable opportunities and optimize their trades, while also allowing investors to model different economic scenarios and market conditions for better

portfolio allocation strategies. Digital twins are useful in retail and ecommerce for creating virtual replicas of products, stores, and supply chains. This capability can contribute to transforming product design and development, streamlining operations, enhancing customer experiences, and driving sales growth. Digital Twins can be used in electricity pricing, auction, and design to optimize energy efficiency, reduce costs, and improve electricity markets. They can help energy analysts detect potential issues and optimize the layout and design of electricity grids to enhance energy efficiency and reduce costs. They can also assist electricity retailers in optimizing bidding strategies in electricity auctions to increase profits and reduce costs. Load forecasting is a crucial application of Digital Twins in electricity pricing, as it enables electricity distributors to accurately anticipate electricity demand and dynamically adjust pricing in real-time to prevent blackouts or brownouts. The digital twin technology involves creating a digital counterpart of a tangible entity, such as a machine, complex systems, or other physical objects. The creation of a digital twin involves utilizing diverse data sources, such as real-time sensor data, historical data, and other relevant information. By integrating this data into a processing system, the digital twin can effectively observe and record the key functionalities of the tangible entity. For instance, if the tangible entity under consideration is a gas turbine, a digital twin of the physical object would be created to mirror its exact specifications, such as size, shape, and technical features. Real-time sensor data from the turbine, including fuel injection rate, air-fuel ratio, inlet air temperature, and exhaust emissions, would be collected and fed into the digital twin. Subsequently, the digital twin would analyze this data and offer insights into the condition monitoring of the gas turbine. The digital twin can be employed to run simulations and analyze performance concerns for a wide range of applications, including fault diagnosis, safety monitoring, and performance optimization. The digital twin technology offers the opportunity to test potential upgrades to a physical object in a virtual environment prior to real-world implementation. This approach provides valuable insights that can be implemented on the physical object, resulting in the ability to improve operational efficiency, minimize downtime, and reduce maintenance expenses. Of particular interest in this work is digital twin technology for forecasting of complex dynamical systems. Forecasting is a critical aspect of digital twin technology as it enables accurate predictions of the behavior of a physical object, enabling proactive maintenance, operational efficiency improvement and safety monitoring. Furthermore, the digital twin can forecast the expected behavior of the physical object in different scenarios, enabling operators to optimize its performance and reduce downtime, while minimizing risks associated with implementing untested changes on the actual physical object. As a result, it is imperative to develop accurate models of physical systems in order to create Digital Twins that can faithfully replicate the behavior of the physical systems for forecasting purposes.

2. Related Work on Time Series Forecasting

Accurately forecasting the behavior of complex dynamical systems, which are characterized by high-dimensional multivariate time series(MTS) in interconnected sensor networks, is crucial for enabling well-informed decision-making in various applications. Forecasting MTS data is challenging due to the intricate relationships among multiple time series variables and the unique features of MTS data, including non-linearity, high-dimensionality and non-stationarity. The spatio-temporal graph neural networks(STGNNs) have become a popular approach to model

the relational dependencies between time series variables in the MTS data for multivariate time series forecasting. Several researchers (e.g., [1, 2, 3, 4, 5, 6]) have contributed to this trend, and their work has significantly advanced the use of GNNs in time series forecasting task. Training STGNNs on the fly is challenging due to their inability to adjust to non-stationary environments and retain past knowledge. The ability of STGNNs to adapt quickly is critical, and successful approaches must handle changes to both new and recurring patterns effectively. However, STGNNs , despite their strong representation learning capabilities, face two major challenges when dealing with time series data streams. Firstly, training STGNNs on data streams in a straightforward manner requires a considerable number of samples to converge. This is because mini-batches or multiple epoch training, commonly used in offline training, are not feasible. Thus, when there is a distribution shift, such neural architectures can become cumbersome and require a large number of samples to learn new concepts effectively, which can ultimately result in suboptimal performance. In essence, the primary challenge lies in the absence of a mechanism within STGNNs to facilitate learning on continuously generated data streams effectively. As a result, the STGNNs must adapt to new trends and patterns in data streams over time. The second challenge arises from the fact that time series data frequently displays recurring patterns that may cease to exist temporarily and then reappear in the future. STGNNs are prone to the catastrophic forgetting phenomenon, whereby the model discards previously acquired knowledge when presented with new data, leading to suboptimal learning of recurring patterns. As a result, this limitation further hinders the overall performance of STGNNs for time series forecasting. Existing STGNNs can learn MTS data dynamics by simultaneously inferring discrete dependency graph structures or by leveraging domain expertise knowledge of predefined relationships among multiple time series variables. While complex dynamical systems consist of interconnected networks, these networks may have higher-order structural relations that extend beyond pairwise associations. Hypergraphs, which provide a more generalized representation of graphs, can effectively model such relations in high-dimensional MTS data. Furthermore, conventional STGNNs prioritize pointwise forecasting and do not offer uncertainty estimates associated with these multi-horizon forecasts. To tackle these challenges, we introduce the Joint Hypergraph Rewiring and Forecasting Neural Framework, which we will refer to as **JHgRF-Net** for brevity. The proposed framework achieves continual learning by balancing two objectives: (i) leveraging prior knowledge to facilitate rapid learning of current trends and patterns, and (ii) maintaining and updating previously acquired knowledge. The **JHgRF-Net** framework achieves dynamic balance between rapid adaptation to recent changes and retrieval of similar old knowledge by leveraging the interaction between two complementary components: the Spatio-Temporal Hypergraph Convolutional Network(STHgCN) and the Spatio-Temporal Transformer Network(STTN). The Mixture of Experts(MOE) approach is utilized to design algorithmic architecture for hypergraph time series forecasting. This approach involves using the aforementioned set of complementary modeling approaches, whose predictions are combined to create a robust mechanism capable of improving the overall accuracy of forecasting. The STHgCN neural operator simultaneously infers discrete dependency hypergraph structure and learns MTS data dynamics. The STHgCN neural operator consists of two sequentially operating modules: hypergraph-structure learning(HgSL) and hypergraph representation learning(HgRL). The HgSL module infers the discrete dependency hypergraph structure and performs hypergraph rewiring to modify the hyperedges so that they better reflect the dependencies between hypernodes. This can involve adding or removing

hyperedges to optimize the relational structure between hypernodes. The HgRL module models the spatio-temporal dynamics underlying the hypergraph-structured MTS data for multi-horizon forecasting. The STTN neural operator learns the underlying dynamics of MTS data beyond the original sparse relational hypergraph structure through a self-attention mechanism. The STTN neural operator learns the underlying dynamics of MTS data beyond the original sparse relational hypergraph structure through a self-attention mechanism. A gating mechanism is utilized to regulate the information flow from complementary components. This mechanism further distills knowledge and improves the accuracy and reliability of the model's predictions. Moreover, the framework captures time-varying uncertainty in forecasts. As a result, the framework provides accurate multi-horizon predictions and reliable uncertainty estimates of forecasts. Furthermore, the framework is designed to provide superior generalization and scalability for large-scale spatio-temporal MTS forecasting tasks that are commonly encountered in real-world applications.

3. Problem Formulation

Let us consider a historical time series dataset with n correlated variables observed over T time steps. The dataset is represented by the notation $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, where the subscript indicates the time step. The observations of all the variables at time step t are denoted by $\mathbf{x}_t = (\mathbf{x}_t^{(1)}, \mathbf{x}_t^{(2)}, \ldots, \mathbf{x}_t^{(n)}) \in \mathbb{R}^{(n \times c)}$, where the superscript refers to the variables. Each sensor can measure multiple physical quantities denoted by c. For example, in intelligent transportation systems, the traffic loop detectors or traffic sensors placed across travel lanes can simultaneously measure three parameters: traffic flow, speed, and volume. Therefore, in this particular case, $c = 3$. In MTSF, we use a rolling-window technique to predict the future values of n-correlated variables for the forecast horizon. At each time step t, we define a look-back window which includes the prior τ-steps of time series data to predict the next v-steps. We use a historical window of n-correlated variables, observed over the previous τ-steps prior to time step t, represented by $\mathbf{X}_{(t-\tau:\ t-1)} \in \mathbb{R}^{n \times \tau \times c}$, to predict the future values of n-variables for the next v-steps, represented by $\mathbf{X}_{(t:t+v-1)} \in \mathbb{R}^{n \times v \times c}$. To capture complex higher-order relationships among variables within the MTS data, we represent the historical data as continuous-time spatial-temporal hypergraphs denoted by $\mathbf{G}_t$. Hypergraphs consist of hypernodes($\mathbf{V}$), representing time series variables and hyperedges($\mathbf{E}$) that capture hierachial relationships among an arbitrary number of hypernodes. The time-dependent hypernode feature matrix is denoted by $\mathbf{X}_{(t-\tau:\ t-1)}$. We learn the implicit hypergraph structure through an embedding-based similarity metric learning approach. The incidence matrix $\mathbf{I} \in \mathbb{R}^{n \times m}$ describes the hypergraph structure, where $\mathbf{I}_{p,q} = 1$ if hyperedge q is incident with hypernode p, and 0 otherwise. Hypergraph sparsity is determined by the number of hyperedges in the hypergraph. In a sparse hypergraph, the number of hyperedges($m(|\mathbf{E}|)$) is relatively small compared to the number of hypernodes($n(|\mathbf{V}|)$), while in a dense hypergraph, the number of hyperedges is relatively large. Sparser hypergraphs generally result in more efficient algorithms, due to the impact of hypergraph sparsity on computational efficiency and algorithmic complexity. A hypergraph with more hyperedges has a denser and more complex structure, resulting in a higher level of connectivity among the hypernodes. Conversely, a hypergraph with fewer hyperedges has a sparser structure with fewer connections between the hypernodes. The proposed framework aims to learn a differentiable function $F(\theta)$ that can predict the future estimates $\mathbf{X}_{(t:t+v-1)}$, of historical window inputs $\mathbf{X}_{(t-\tau:\ t-1)}$, given a hypergraph $\mathbf{G}_t$. To put it

briefly, the function $F(\theta)$ takes in the past observations and hypergraph structure, represented by $[\mathbf{x}_{(t-\tau)}, \cdots, \mathbf{x}_{(t-1)}; \mathbf{G}_t]$, and predict future observations, denoted as $[\mathbf{x}_{(t+1)}, \cdots, \mathbf{x}_{(t+v-1)}]$. This is mathematically represented as:

$$\left[\mathbf{x}_{(t-\tau)}, \cdots, \mathbf{x}_{(t-1)}; \mathbf{G}_t\right] \xrightarrow{F(\theta)} \left[\mathbf{x}_{(t+1)}, \cdots, \mathbf{x}_{(t+v-1)}\right]$$

The MTSF task formulated on the implicit hypergraph($\mathbf{G}_t$), can be expressed as shown below:

$$\min_{\theta} \mathcal{L}_{\text{MAE}}\left(\mathbf{X}_{(t:t+v-1)}, \widehat{\mathbf{X}}_{(t:t+v-1)}; \mathbf{X}_{(t-\tau:\ t-1)}, \mathbf{G}_t\right)$$

The function $F(\theta)$ involves a set of parameters θ which can be trained to optimize its performance. The predicted future observations is denoted by $\widehat{\mathbf{X}}_{(t:t+v-1)}$. To train the learning algorithm, we minimize the loss function denoted by $\mathcal{L}_{\text{MAE}}$, i.e., the mean absolute error(MAE), which is defined as:

$$\mathcal{L}_{\text{MAE}} = \frac{1}{v}\left|\mathbf{X}_{(t:t+v-1)} - \widehat{\mathbf{X}}_{(t:t+v-1)}\right|$$

Here, $\mathbf{X}_{(t:t+v-1)}$ is the actual future MTS data, and $\frac{1}{v}$ is a scaling factor.
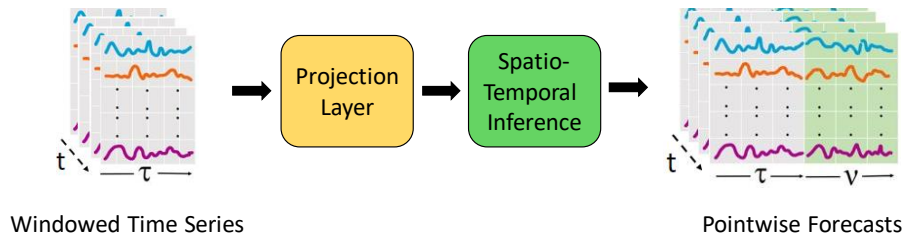
Figure 1: Overview of **JHgRF-Net** framework.

4. OUR APPROACH

Our proposed neural forecasting framework consists of two key components: the projection layer and the spatio-temporal feature extractor, as shown in Figure 1. The spatio-temporal inference component includes two distinct methods for hypergraph representation learning: the Spatio-Temporal Hypergraph Convolutional Network(STHgCN) and the Spatio-Temporal Transformer Network(STTN). The STHgCN method employs hypergraph as a mathematical model for learning the underlying higher-order relations of the time series variables. This is achieved by optimizing the discrete hypergraph structure underlying the observed data. It then peforms the gated hypergraph convolution operations on the hypergraph-structured MTS data to model the intricate spatio-temporal dynamics within the latent hypernode-level representations. The final representations can then be used to predict multi-horizon forecasts. The STTN method is a powerful technique for modeling the hypergraph-structured MTS data. The STTN method extends transformer networks to handle arbitrary sparse hypergraph structures with full attention as a useful inductive bias. This enables the model to learn intra- and inter-correlations among the variables without being limited by the hierarchical structural information underlying the MTS data. It leverages task-specific relations between variables beyond the original sparse structure to generate expressive hypernode-level representations that improve forecast accuracy. We use a gating mechanism to regulate the flow of information from the two methods. This enables us to learn optimal representations of the hypernode-level representations that capture the accurate

dynamics of complex interconnected sensor networks. To summarize, our framework performs the joint optimization of the different learning components to generate accurate forecasts across multiple forecast horizons, while also ensuring reliable estimates of uncertainty for time-series forecasting tasks.
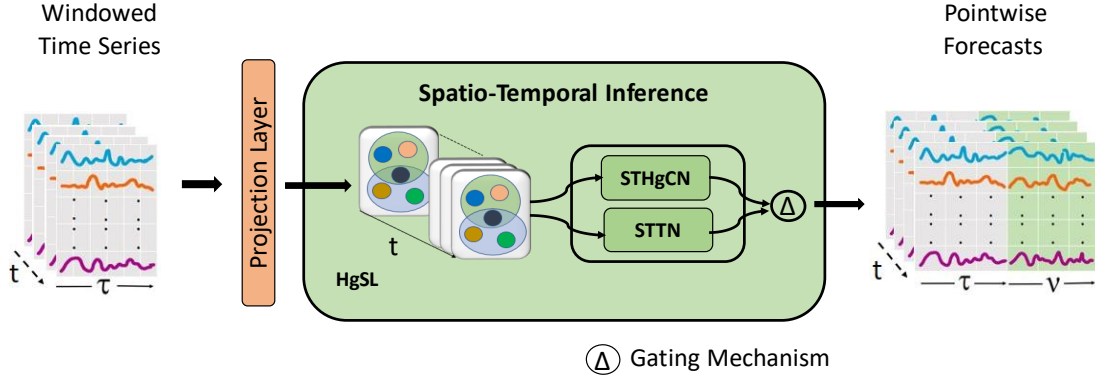
Figure 2: The above figure illustrates the **JHgRF-Net** framework, which jointly learns to incorporate a discrete dependency hypergraph structure between multiple time series variables in order to capture complex interactions. The hypergraph representation learning algorithm is used to model intrinsic spatio-temporal dependencies for multi-horizon forecasting.

4.1. PROJECTION LAYER

The proposed framework uses a projection layer with gated linear networks(GLN, [7]) to obtain non-linear representations of input data. Specifically, the input data $\mathbf{X}_{(t-\tau:\ t-1)} \in \mathbb{R}^{n \times \tau \times c}$ is transformed through a gating mechanism, resulting in $\overline{\mathbf{X}}_{(t:t+v-1)} \in \mathbb{R}^{n \times v \times d}$, which represents the non-linear transformed input data. It is described as follows:

$$\overline{\mathbf{X}}_{(t:t+v-1)} = \big(\sigma(\mathbf{W}_0 \mathbf{X}_{(t-\tau:\ t-1)}) \otimes \mathbf{W}_1 \mathbf{X}_{(t-\tau:\ t-1)}\big)\mathbf{W}_2$$

Here, the trainable weight matrices are $\mathbf{W}_0, \mathbf{W}_1 \in \mathbb{R}^{c \times d}, \mathbf{W}_2 \in \mathbb{R}^{\tau \times v}$, and the element-wise multiplication is denoted by $\otimes$. The utilization of a non-linear activation function σ improves representation learning and enables the framework to effectively learn and model complex patterns present in the MTS data.

4.2. SPATIAL-INFERENCE

Figure 2 illustrates the spatio-temporal feature extractor of the framework, which consists of two distinct methods(STHgCN and STTN). Further information regarding each method will be elaborated in the subsequent sections.

4.2.1. Spatio-Temporal Hypergraph Convolutional Network(STHgCN)

The STHgCN method comprises sequentially operating modules, including hypergraph structure learning(HgSL) and hypergraph representation learning(HgRL) modules. The following sections will elaborate on each module and provide more details.

4.2.1.1. Implicit hypergraph Inferenece

The HgSL module uses an embedding-based similarity metric learning technique to capture higher-order dependency relationships between different variables in the MTS data and computes an optimal discrete hypergraph structure for a hypergraph-structured representation of the MTS data. In short, the implicit hypergraph provides a spatio-temporal inductive bias that enables a structured representation of the MTS data, capturing the underlying relationships and dependencies among the variables. The hypernodes and hyperedges of the hypergraph are represented by the differentiable embeddings in the d-dimensional vector space, $\mathbf{z_i}, \mathbf{z_j} \in \mathbb{R}^{(d)}$, where $1 \leq i \leq n$ and $1 \leq j \leq m$. By leveraging the learned embeddings to transform the MTS data into a hypergraph-structured time series data, the HgSL module computes the optimal hypergraph topology that captures the task-relevant relationships and dependencies among the variables, making it a powerful tool for learning relational hypergraph structures from complex MTS data. The pairwise similarity($P_{i,j}$) between any pair of $\mathbf{z_i}$ and $\mathbf{z_j}$ is computed as follows:

$$P_{i,j} = \sigma\big([S_{i,j}||1 - S_{i,j}]\big); S_{i,j} = \frac{\mathbf{z_i^T z_j} + 1}{2\,\|\mathbf{z_i}\| \cdot \|\mathbf{z_j}\|}$$

where $\|$ denotes vector concatenation. The differentiable, sigmoid activation function is applied to map the pairwise scores to the interval [0,1]. The hyperedge probability over hypernodes of the hypergraph is represented as $P_{i,j}^{(k)} \in \mathbb{R}^{nm \times 2}$, where $k \in \{0, 1\}$. The scalar value of $P_{i,j}^{(k)} \in [0, 1]$ indicates the relationship between a pair of hypernodes and hyperedges, indexed by (i, j). To be precise, $P_{i,j}^{(0)}$ represents the probability of hypernode i being connected to hyperedge j, while $P_{i,j}^{(1)}$ denotes the probability that hypernode i is not connected to hyperedge j. To accurately and efficiently sample discrete hypergraph structures from the hyperedge probability distribution $P_{i,j}$, we leverage the Gumbel-softmax trick introduced in [8]. This technique is powerful in capturing complex relationships among variables in MTS data, making the HgSL module more effective. The connectivity pattern of the hypergraph structure is then represented using an incidence matrix $\mathbf{I} \in \mathbb{R}^{n \times m}$, which captures the relationships between hypernodes and hyperedges in the hypergraph. By using the Gumbel-softmax trick, we can learn the hypergraph structure in an end-to-end differentiable manner. Thus, it becomes possible to apply the gradient-based optimization methods during model training, enabling an inductive-learning approach to learn complex underlying structures within the MTS data. The Gumbel-Softmax trick involves using random noise from the Gumbel distribution to perturb the hyperedge probability distribution and then sampling the optimal discrete structure from the distribution using the Softmax function. The incidence matrix is obtained as,

$$\mathbf{I}_{i,j} = \exp\big((g_{i,j}^{(k)} + P_{i,j}^{(k)} + \epsilon)/\gamma\big)\Big/\sum \exp\big((g_{i,j}^{(k)} + P_{i,j}^{(k)} + \epsilon)/\gamma\big)$$

where the temperature parameter(γ) of Gumbel-Softmax trick is set to 0.05, and ϵ is a small constant added to avoid numerical instability. Random noise, denoted by $g_{ij}^{(k)} \sim \mathrm{Gumbel}(0, 1) = \log(-\log(U(0, 1))$ is sampled from the Gumbel distribution, where U represents the uniform distribution with a range of 0 to 1. We optimize the hypergraph distribution parameters to ensure that the learned hypergraph is sparse, eliminating redundant hyperedges over hypernodes. The forecasting task provides indirect supervisory information that helps to reveal the hypergraph relation structure in the observed MTS data. In summary, the HgSL module learns the latent hypergraph structure of multiple interacting time series variables to create a structured

representation of the time series data, which facilitates downstream multi-horizon forecasting with predictive uncertainty estimation.

4.2.1.2. Hypergraph Attention Network(HgAT)

The HgAT neural operator extends attention-based convolution operations to non-Euclidean domains, such as hypergraphs. It accurately models the complex hypergraph-structured MTS data, thereby improving multi-horizon forecast accuracy. The HgAT operator captures spatial correlations among time-series variables by encoding relational inductive bias within the hypergraph's connectivity. It performs message-passing schemes to propagate information through the hypergraph-structured MTS data, which is characterized by an incidence matrix (represented by $\mathbf{I} \in \mathbb{R}^{n \times m}$) and a feature matrix (represented by $\overline{\mathbf{X}}_{(t:t+v-1)} \in \mathbb{R}^{n \times v \times d}$) to compute the hypernode representation matrix (represented by $\mathbf{H}_{(t:t+v-1)} \in \mathbb{R}^{n \times v \times d}$). Each row in the matrix $\mathbf{H}_{(t:t+v-1)}$ represents the hypernode representations, $\mathbf{h}_i^t \in \mathbb{R}^{v \times d}$. The HgAT operator captures relationships among time-series variables by encoding structural and feature characteristics of spatio-temporal hypergraphs in hypernode representations. It adapts to changes in time-series variable dependencies over time in the hypernode representations $\mathbf{h}_i^t$. The HgAT operator models spatio-temporal correlations among time-series variables in hypergraph-structured MTS data using intra-edge and inter-edge neighborhood aggregation schemes. The intra-edge aggregation considers hypernodes associated with a specific hyperedge, while inter-edge aggregation considers hyperedges connected to a specific hypernode. In hypergraph-structured MTS data, hyperedges capture relationships between multiple time-series variables, which can have varying degrees of correlation and complexity. Let the notation $\mathbf{N}_{j,i}$ represent a subset of hypernodes i associated with a specific hyperedge j. The intra-edge neighborhood of a hypernode i, denoted as $\mathbf{N}_{j,i} \backslash i$, captures a localized cluster of semantically-corelated time-series variables and their higher-order relationships. The inter-edge neighborhood of a hypernode i, represented by $\mathbf{N}_{i,j}$, includes the set of hyperedges j connected to that hypernode, providing a more comprehensive understanding that each variable may have multiple and potentially complex relationships with other time-series variables in the data. We use attention-based intra-edge neighborhood aggregation to obtain latent hyperedge representations, which leads to a more comprehensive understanding of the MTS data. This approach can be described as follows:

$$\mathbf{h}_j^{(t,\ell)} = \sum_{z=1}^{\mathbf{Z}} \sigma \Big(\sum_{i \in \mathbf{N}_{j,i}} \alpha_{j,i}^{(t,\ell,z)} \mathbf{W}_0^{(z)} \mathbf{h}_i^{(t,\ell-1,z)} \Big)$$

where the hyperedge representations at layer ℓ are denoted by $\mathbf{h}_j^{(l)} \in \mathbb{R}^{v \times d}$. Each hypernode's initial representation is its corresponding feature vector,

$$\mathbf{h}_i^{(t,0,z)} = \overline{\mathbf{x}}_i^{(t)}$$

where $\overline{\mathbf{x}}_i^{(t)} \in \mathbb{R}^{v \times d}$ represents the ith row of the feature matrix $\overline{\mathbf{X}}_{(t:t+v-1)} \in \mathbb{R}^{n \times v \times d}$. At each layer, the HgAT operator produces multiple representations denoted by $\mathbf{h}_j^{(l,z)}$ of the input data, each with its own set of parameters, and combines them by summation. This enables the HgAT operator to capture various aspects of the relations underlying the intra-edge neighborhood in the hypergraph-structured MTS data. To determine the attention coefficient $\alpha_{j,i}$ for the hypernode i incident with hyperedge j, we compute its relative importance as follows:

$$e_{j,i}^{(t,\ell,z)} = \text{ReLU}\left(\mathbf{W}_0^{(z)}\mathbf{h}_i^{(t,\ell-1,z)}\right)$$

$$\alpha_{j,i}^{(t,\ell,z)} = \frac{\exp\left(e_{j,i}^{(t,\ell,z)}\right)}{\sum_{k \in \mathbf{N}_{j,i} \cup i} \exp\left(e_{j,k}^{(t,\ell,z)}\right)}$$

where $e_{j,i}$ denotes the unnormalized attention score. The HgAT method utilizes an attention-based inter-edge neighborhood aggregation scheme, which captures complex dependencies and relationships between hyperedges and hypernodes. It generates expressive hypernode representations by summing over ReLU activations of linear transformations of previous layer hypernode representations and weighted hyperedge representations. This is described below,

$$\mathbf{h}_i^{(t,\ell)} = \sum_{z=1}^{\mathbf{Z}} \text{ReLU}\left(\mathbf{W}_0^{(z)}\mathbf{h}_i^{(t,\ell-1,z)} + \sum_{j \in \mathbf{N}_{i,j}} \beta_{i,j}^{(t,\ell,z)}\mathbf{W}_1^{(z)}\mathbf{h}_j^{(t,\ell,z)}\right)$$

The weight matrices that are trained are represented as $\mathbf{W}_0^{(z)}, \mathbf{W}_1^{(z)} \in \mathbb{R}^{d \times d}$. The ReLU activation function is used to introduce non-linearity while updating the hypernode-level representations. The attention scores $\beta_{i,j}$ are normalized and determine the relevance of each hyperedge j that is incident with hypernode i. This allows the HgAT operator to focus on the most significant hyperedges, and the attention scores are computed as follows:

$$\phi_{i,j}^{(t,\ell,z)} = \text{ReLU}\left(\mathbf{W}_3^{(z)} \cdot \left(\mathbf{W}_2^{(z)}\mathbf{h}_i^{(t,\ell-1,z)} \oplus \mathbf{W}_2^{(z)}\mathbf{h}_j^{(t,\ell,z)}\right)\right)$$

$$\beta_{i,j}^{(t,\ell,z)} = \frac{\exp(\phi_{i,j}^{(t,\ell,z)})}{\sum_{k \in \mathbf{N}_{i,j} \cup j} \exp(\phi_{i,k}^{(t,\ell,z)})}$$

where $\mathbf{W}_2^{(z)} \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_3^{(z)} \in \mathbb{R}^{2d}$ are trainable weight matrix and vector, respectively. $\oplus$ denotes the concatenation operator. The unnormalized attention score is denoted by $\phi_{i,j}$. Batch normalization and dropout techniques are used to enhance generalization and mitigate overfitting. A gating mechanism is employed to selectively combine features from $\overline{\mathbf{x}}_i^{(t)}$ and $\mathbf{h}_i^{(t,\ell)}$ in a differentiable way. These methods improve the HgAT operator reliability and accuracy for the downstream MTSF task.

$$g^{(t)} = \sigma\left(f_s(\mathbf{h}_i^{(t,\ell)}) + f_g(\overline{\mathbf{x}}_i^{(t)})\right)$$

$$\mathbf{h}_i^{(t,\ell)} = \sigma\left(g^{(t)}(\mathbf{h}_i^{(t,\ell)}) + (1 - g^{(t)})(\overline{\mathbf{x}}_i^{(t)})\right)$$

where f_s and f_g denote the linear projections, enabling the HgAT operator to capture the relationships between time-series variables and their temporal changes, resulting in enhanced forecast accuracy. In summary, the HgAT operator is a powerful technique for encoding and analyzing spatio-temporal hypergraphs.

4.2.1.3. Saptio-temporal Hypergraph Representation Learning

We present the spatio-temporal hypergraph representation learning(HgRL) module to operate on a sequence of dynamic hypergraphs, where hypergraph structure is fixed, and hypernode attributes change over time, where each hypergraph represents the hypergraph-structured MTS data at a specific time step. The HgRL operator utilizes Gated Recurrent Units(GRU, [9]) to model the spatio-temporal dynamics of the dynamic hypergraph sequence. The computation of the update gate, reset gate, and hidden state in a traditional GRU involves matrix multiplication with

weight matrices. In the HgRL module, however, these matrix multiplications are replaced with Hypergraph Attention Networks(HgAT). The HgRL operator analyzes hypergraph-structured MTS data over time. It propagates information between hypernodes across different time steps, which enables the model to capture the complex spatio-temporal dependencies between the hypergraphs. The HgRL operator utilizes the implicit hypergraph topology to propagate information between hypernodes by averaging the hypernode representations in their local neighborhood at each time step computed as follows,

$$\mathbf{U}_{t:t+v-1} = \sigma \left(\mathbf{W}_u \left[f\left(\mathbf{I}, \mathbf{X}_{(t:t+v-1)}\right) \| \mathbf{H}_{t-\tau:\, t-1} \right] + \mathbf{B}_u \right)$$

$$\mathbf{R}_{t:t+v-1} = \sigma \left(\mathbf{W}_r \left[f\left(\mathbf{I}, \mathbf{X}_{(t:t+v-1)}\right) \| \mathbf{H}_{t-\tau:\, t-1} \right] + \mathbf{B}_r \right)$$

$$\mathbf{C}_{t:t+v-1} = \tanh \left(\mathbf{W}_c \left[f\left(\mathbf{I}, \mathbf{X}_{(t:t+v-1)}\right) \| \left(\mathbf{R}_{t:t+v-1} \otimes \mathbf{H}_{t-\tau:\, t-1}\right) \right] + \mathbf{B}_c \right)$$

$$\mathbf{H}_{t:t+v-1} = \mathbf{U}_{t:t+v-1} \otimes \mathbf{H}_{t-\tau:\, t-1} + \left(1 - \mathbf{U}_{t:t+v-1}\right) \otimes \mathbf{C}_{t:t+v-1}$$

where $f\left(\mathbf{I}, \mathbf{X}_{(t:t+v-1)}\right)$ denote the HgAT operator. $\|$, and $\otimes$ denotes the concatenation operation and element-wise multiplication operation. The update and reset gates at time t are represented by the matrices $\mathbf{U}_{t:t+v-1}$ and $\mathbf{R}_{t:t+v-1}$, respectively. $\mathbf{W}_r, \mathbf{W}_u$, and $\mathbf{W}_c$ are learnable weight matrices and $\mathbf{B}_u, \mathbf{B}_r$, and $\mathbf{B}_c$ are learnable biases. In summary, the node representation matrix, $\mathbf{H}_{t:t+v-1}$ captures the spatio-temporal dynamics at different scales underlying the discrete-time dynamic hypergraphs, where each row in $\mathbf{H}_{t:t+v-1}$ represents the hypernode representations $\mathbf{h_i}^{(t)} \in \mathbb{R}^{v \times d}, \forall i \in \mathbf{V}$. Some of the key advantages of T-HGCN operator over traditional methods include its ability to handle large and sparse spatio-temporal hypergraphs. The STHgCN method utilizes useful relational inductive bias encoded in the hypergraph-structured data for modeling the continuous-time nonlinear dynamics of the complex system to disentangle the various latent aspects underneath the data for better forecast accuracy.

4.2.2. Spatio-Temporal Transformer Network(STTN)

The Spatio-temporal transformer network(STTN) operator is a new extension of transformer networks ([10]) that incorporates full attention as a desired inductive bias to model MTS data with arbitrary sparse hypergraph structures. This capability enables it to capture fine-grained spatio-temporal dependencies in MTS data, unconstrained by hierarchical structural information underlying the MTS data. By allowing attention to all hypernodes within the hypergraph, the neural operator can span large receptive fields and reason globally about complex dependencies in hypergraph-structured MTS data. As a result, it can serve as a drop-in replacement for existing methods that model hierarchical relationships among time-series variables in MTS data. Additionally, the neural operator is particularly suitable for downstream forecasting tasks in spatio-temporal hypergraphs. The transformer encoder comprises alternating layers of multi-headed self-attention(MSA) and multi-layer perceptron(MLP) blocks to capture both local and global contextual information. To enhance performance and regularize the transformer operator, each block is followed by layer normalization(LN([11])) and residual connections. The skip-connections are incorporated through an initial connection strategy inspired by ResNets([12]) to address vanishing gradients and over-smoothing issues and enable the learning of complex and deep representations of the data. Using a space-then-time(STT, [13]) approach, the STTN first performs a temporal-encoding step to capture the long-term temporal dependencies(intra-dependencies) within the time series variables. This is followed by a spatial-encoding step,

which captures the inter-dependencies among the time series variables. We model the intra- and inter-dependencies through a sequential operating temporal and spatial transformer networks, respectively.

4.2.2.1. Temporal Transformer

In self-attention mechanism, the input sequences is transformed into three tensors: the query tensor, the key tensor, and the value tensor, where the input tensor is denoted by $\overline{\mathbf{X}}_{(t:t+v-1)} \in \mathbb{R}^{n \times v \times d}$ and has three dimensions: number of time series variables(n), forecast horizon(v), and embedding dimension(d). The key tensor is searched using the query tensor to retrieve relevant information, and the value tensor is weighted by the resulting attention weights. The weighted value tensor is then summed to produce the final output. To begin with, we reshape the input tensors to split the embedding dimension into multiple heads:

$$\text{queries}_{n,v,d}^{(t:t+v-1)} = \text{queries}_{n,q,h*h_d}^{(t:t+v-1)} \rightarrow \text{queries}_{n,q,h,h_d}^{(t:t+v-1)}$$

$$\text{keys}_{n,v,d}^{(t:t+v-1)} = \text{keys}_{n,k,h*h_d}^{(t:t+v-1)} \rightarrow \text{keys}_{n,k,h,h_d}^{(t:t+v-1)}$$

$$\text{values}_{n,v,d}^{(t:t+v-1)} = \text{values}_{n,k,h*h_d}^{(t:t+v-1)} \rightarrow \text{values}_{n,k,h,h_d}^{(t:t+v-1)}$$

where, h and h_d represents the index of the attention head, and head dimension, respectively. Here, q and k represent the indices of the query and key positions, respectively. We compute the energy between queries and keys, as described below.

$$\text{energy}_{n,q,k,h}^{(t:t+v-1)} = \sum_{h_d} \text{queries}_{n,q,h,h_d}^{(t:t+v-1)} \cdot \text{keys}_{n,k,h,h_d}^{(t:t+v-1)}$$

We compute the attention scores using the softmax function described below:

$$\text{attention}_{n,q,k,h}^{(t:t+v-1)} = \frac{\exp\left(\text{energy}_{n,q,k,h}^{(t:t+v-1)} / \sqrt{h_d}\right)}{\sum_{k'} \exp\left(\text{energy}_{n,q,k',h}^{(t:t+v-1)} / \sqrt{h_d}\right)}$$

To calculate the output tensor, we multiply the values tensor with the attention scores, which is described below,

$$\text{out}_{n,q,h,h_d}^{(t:t+v-1)} = \sum_{k} \text{attention}_{n,q,k,h_d}^{(t:t+v-1)} \cdot \text{values}_{n,k,h,h_d}^{(t:t+v-1)}$$

We perform the concatenation operation along the h dimension, which combines the outputs of all the heads. We apply a linear transformation to obtain the final output, as follows,

$$out_{n,v,d}^{(t:t+v-1)} = \text{out}_{n,q,h*h_d}^{(t:t+v-1)} \mathbf{W}_{h*h_d,d}$$

4.2.2.2. Spatial Transformer

The output of the temporal transformer, denoted by $\text{out}^{(t:t+v-1)} \in \mathbb{R}^{(n \times v \times d)}$, is passed to the spatial transformer as input, and it consists of three dimensions: number of time series variables(n), forecast horizon(v), and embedding dimension(d). The input sequences are first transformed to three tensors, namely the query tensor, the key tensor, and the value tensor, before applying the self-attention mechanism. In order to retrieve relevant information, the query tensor is employed

to search through the key tensor. The resulting attention scores are then used to weight the value tensor. Finally, the weighted values are subsequently aggregated to produce the final output. We reshape the input tensors to split the embedding dimension into multiple heads:

$$\text{queries}_{q,v,d}^{(t:t+v-1)} = \text{queries}_{q,v,h*h_d}^{(t:t+v-1)} \rightarrow \text{queries}_{q,v,h,h_d}^{(t:t+v-1)}$$

$$\text{keys}_{k,v,d}^{(t:t+v-1)} = \text{keys}_{k,v,h*h_d}^{(t:t+v-1)} \rightarrow \text{keys}_{k,v,h,h_d}^{(t:t+v-1)}$$

$$\text{values}_{k,v,d}^{(t:t+v-1)} = \text{values}_{k,v,h*h_d}^{(t:t+v-1)} \rightarrow \text{values}_{k,v,h,h_d}^{(t:t+v-1)}$$

where, h and h_d denote the number of heads, and head dimension, respectively. We compute the energy between queries and keys, as described below.

$$\text{energy}_{q,k,v,h}^{(t:t+v-1)} = \sum_{h_d} \text{queries}_{q,v,h,h_d}^{(t:t+v-1)} \cdot \text{keys}_{k,v,h,h_d}^{(t:t+v-1)}$$

We obtain the attention scores using the softmax function:

$$\text{attention}_{q,k,v,h}^{(t:t+v-1)} = \frac{\exp\left(\text{energy}_{q,k,v,h}^{(t:t+v-1)} / \sqrt{h_d}\right)}{\sum_{k'} \exp\left(\text{energy}_{q,k',v,h}^{(t:t+v-1)} / \sqrt{h_d}\right)}$$

We then compute the output tensor by multiplying the attention scores with the values tensor:

$$\text{out}_{q,v,h,h_d}^{(t:t+v-1)} = \sum_{k} \text{attention}_{q,k,v,h}^{(t:t+v-1)} \cdot \text{values}_{k,v,h,h_d}^{(t:t+v-1)}$$

We apply a linear transformation to obtain the final output, as follows,

$$\mathbf{out}_{n,v,d}^{(t:t+v-1)} = out_{n,v,h*h_d}^{(t:t+v-1)} \mathbf{W}_{h*h_d,d}$$

4.2.3. Gating Mechanism

The mixture-of-experts(MOE) mechanism in deep learning combines predictions from multiple subnetworks, such as "STHgCN" and "STTN" representation learning methods, through a gating mechanism that computes a weighted sum of their predictions based on the input. The aim is to find the optimal weight assignment for the gating function and train the experts accordingly using these weights. From a cooperative game theory perspective, the MOE is a cooperative game where experts collaborate to optimize the system's overall performance. The gating mechanism can optimize the weights assigned to each expert by evaluating their individual performance, as well as the system's overall performance. The fused representations in MOE are obtained by combining expert predictions using the gating mechanism weights. This is described below:

$$g'' = \sigma\left(f_s''(\mathbf{H}_{t:t+v-1}) + f_g''(\mathbf{out}^{(t:t+v-1)})\right)$$

$$\widehat{\mathbf{X}}_{(t:t+v-1)} = \sigma\left(g''(\mathbf{H}_{t:t+v-1}) + (1 - g'')(\mathbf{out}^{(t:t+v-1)})\right)$$

where, $\widehat{\mathbf{X}}_{(t:t+v-1)}$ are model multi-horizon forecasts. $\mathbf{H}_{t:t+v-1}$ and $\mathbf{out}_{(t:t+v-1)}$ denote the hypernode representation matrix computed by the STHgCN and STTN neural network

methods, respectively. f_s'' and f_g'' are linear projections. Moreover, our framework variant(**w/Unc-JHgRF-Net**) ensures precise and reliable uncertainty estimates of multi-horizon forecasts by minimizing the negative Gaussian log likelihood. For more details, refer to the appendix. The proposed methods(**JHgRF-Net**, **w/Unc-JHgRF-Net**) enable end-to-end modeling of hidden interdependencies and their evolution over time in sensor network-based dynamical systems for highly accurate forecasting task.

5. Datasets

The study aims to evaluate the effectiveness of two new models, **JHgRF-Net** and **w/Unc-JHgRF-Net**(**JHgRF-Net** with local-uncertainty estimation), on large-scale spatial-temporal datasets([14]) containing real-world traffic information. The datasets include PeMSD3, PeMSD4, PeMSD7, PeMSD7(M), and PeMSD8. The study includes a preprocessing step to ensure consistency with prior research by aggregating the 30-second interval data into 5-minute averages. Additionally, publicly accessible METR-LA and PEMS-BAY datasets([15]) were used for traffic flow prediction. The preprocessing step involves transforming the time series data into 5-minute interval averages to ensure a fair comparison with the prior research. For all the above-mentioned traffic datasets, we possess information about the underlying sensor graph. To create the sensor graph, we calculated the distances between sensors in the road network and utilized a thresholded Gaussian kernel to build the adjacency matrix. Our experimental findings, discussed in the next section, support the rationale of learning the implicit hypergraph relational structure of the variables underlying the MTS data and modeling the spatial-temporal dynamics for improved forecast accuracy compared to the learning to forecast on predefined(prior-known) sensor graphs.Furthermore, we utilize various multivariate datasets, including Electricity[1], Solar-energy[2], Exchange-rate[3], and Traffic[4], for which no prior sensor graph structure exists. Additionally, the SWaT([16]) and WADI([17]) are sensor datasets that measure water treatment plants and also do not have a predefined sensor graph structure. They were first used in prior research for anomaly detection due to the presence of annotated anomalies, but later used in forecasting experiments because their training sets are anomaly-free. The experimental study conducted on benchmark datasets aims to showcase the effectiveness and advantages of the proposed methodology(**JHgRF-Net** and **w/Unc-JHgRF-Net**) in analyzing and modeling complex spatio-temporal MTS data, surpassing existing methods.

6. Experimental results

Table 2 provides a thorough comparison between the proposed models(**JHgRF-Net** and **w/Unc-JHgRF-Net**), and several baseline models on the MTSF task across five different benchmark datasets: PeMSD3, PeMSD4, PeMSD7, PeMSD7M, and PeMSD8. To evaluate the models effectiveness, we measured forecast errors for a well-established benchmark, involving a $12(\tau)$-step-prior to $12(\upsilon)$-step-ahead forecasting task. We utilize a multi-metric approach in forecasting tasks to comprehensively evaluate the proposed models performance compared to the baseline models. We use several performance metrics, including mean absolute error(MAE), root mean

---

[1]archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

[2]www.nrel.gov/grid/solar-power-data.html

[3]github.com/laiguokun/multivariate-time-series-data

[4]https://pems.dot.ca.gov

Table 1
The pointwise forecast errors on benchmark datasets for multi-horizon prediction tasks.

	Model	MAE				Model	MAE		
		Horizon @ 3	Horizon @ 6	Horizon @ 12			Horizon @ 3	Horizon @ 6	Horizon @ 12
METR-LA	LSTM	3.495 ± 0.010	3.712 ± 0.012	4.105 ± 0.011	**SWaT**	LSTM	0.300 ± 0.013	0.329 ± 0.011	0.432 ± 0.016
	LSTM-U	3.416 ± -	4.092 ± -	5.141 ± -		LSTM-U	0.287 ± 0.001	0.487 ± 0.001	0.883 ± 0.001
	NRI	4.680 ± 0.081	6.388 ± 0.087	8.466 ± 0.099		NRI	0.415 ± 0.014	0.479 ± 0.014	0.641 ± 0.011
	GDN	3.149 ± 0.017	3.482 ± 0.014	3.909 ± 0.012		GDN	0.803 ± 0.045	0.854 ± 0.069	1.081 ± 0.149
	MTGNN	3.016 ± 0.004	3.574 ± 0.005	4.308 ± 0.006		MTGNN	0.488 ± 0.011	0.537 ± 0.016	0.704 ± 0.025
	GTS	2.884 ± 0.005	3.269 ± 0.006	**3.701** ± 0.006		GTS	0.242 ± 0.038	0.279 ± 0.039	0.387 ± 0.044
	JHgRF-Net	**2.039** ± 0.010	**2.059** ± 0.02	**4.937** ± 0.015		**JHgRF-Net**	**0.074** ± 0.006	**0.136** ± 0.012	**0.170** ± 0.003
	w/Un-JHgRF-Net	2.066 ± 0.020	3.287 ± 0.040	5.150 ± 0.033		w/Un-JHgRF-Net	1.183 ± 0.169	0.184 ± 0.318	±
PEMS-BAY	LSTM	2.043 ± 0.005	2.111 ± 0.005	2.242 ± 0.007	**Traffic**	LSTM	0.016 ± 0.0002	0.018 ± 0.0003	0.017 ± 0.0003
	GDN	1.890 ± 0.009	2.021 ± 0.009	2.172 ± 0.012		LSTM-U	0.029 ± 0.0001	0.033 ± 0.0002	0.029 ± 0.0001
	MTGNN	1.319 ± 0.002	1.690 ± 0.002	2.101 ± 0.004		NRI	0.013 ± 0.0001	0.013 ± 0.0001	0.014 ± 0.0001
	GTS	1.268 ± 0.0002	1.555 ± 0.001	1.813 ± 0.003		MTGNN	0.010 ± 0.0003	0.011 ± 0.0004	0.011 ± 0.0003
	JHgRF-Net	**0.806** ± 0.0003	**1.217** ± 0.004	**1.758** ± 0.001		**JHgRF-Net**	**0.006** ± 0.0001	**0.009** ± 0.0001	**0.011** ± 0.0001
	w/Un-JHgRF-Net	0.801 ± 0.001	1.212 ± 0.017	1.767 ± 0.0001		w/Un-JHgRF-Net	0.030 ± 0.0001	0.030 ± 0.0001	0.030 ± 0.0001
WADI	LSTM	6.658 ± 0.055	6.755 ± 0.065	6.789 ± 0.041	**Electricity**	LSTM	323.345 ± 3.854	384.239 ± 10.789	352.488 ± 4.217
	LSTM-U	5.980 ± 0.036	6.106 ± 0.023	6.351 ± 0.052		LSTM-U	710.917 ± 0.732	1079.394 ± 2.999	849.250 ± 1.895
	GDN	7.409 ± 0.307	7.442 ± 0.221	7.523 ± 0.282		GDN	265.166 ± 3.081	269.224 ± 2.367	280.400 ± 1.447
	MTGNN	5.953 ± 0.038	6.108 ± 0.042	6.254 ± 0.0293		MTGNN	170.155 ± 2.977	186.004 ± 4.751	193.499 ± 4.465
	GTS	5.474 ± 0.007	5.575 ± 0.009	5.772 ± 0.008		GTS	175.877 ± 1.022	**185.791** ± 0.931	199.583 ± 1.375
	JHgRF-Net	**4.149** ± 0.029	**4.592** ± 0.208	**4.758** ± 0.118		**JHgRF-Net**	**160.955** ± 0.136	178.781 ± 1.522	225.724 ± 2.109
	w/Un-JHgRF-Net	4.283 ± 0.011	4.773 ± 2.203	4.936 ± 0.076		w/Un-JHgRF-Net	145.201 ± 0.504	214.605 ± 0.641	±
Solar Energy	LSTM	1.982 ± 0.016	2.677 ± 0.021	4.241 ± 0.023	**Exchange rate**	LSTM	0.0141 ± 0.0013	0.0187 ± 0.0020	0.0190 ± 0.0018
	LSTM-U	2.760 ± 0.007	4.366 ± 0.003	6.282 ± 0.002		LSTM-U	0.0057 ± 0.0002	0.0076 ± 0.0001	0.0102 ± 0.0001
	GDN	2.095 ± 0.018	2.329 ± 0.021	2.855 ± 0.040		NRI	0.0047 ± 0.0001	0.0073 ± 0.0002	0.0111 ± 0.0005
	MTGNN	1.512 ± 0.005	2.051 ± 0.008	2.689 ± 0.013		MTGNN	0.0109 ± 0.0023	0.0146 ± 0.0033	0.0136 ± 0.0013
	GTS	1.419 ± 0.004	1.926 ± 0.013	2.657 ± 0.033		GTS	**0.0047** ± 0.0000	0.0070 ± 0.0000	0.0099 ± 0.0000
	JHgRF-Net	**0.575** ± 0.013	**0.868** ± 0.035	**0.873** ± 0.034		**JHgRF-Net**	**0.0043** ± 0.0001	**0.0044** ± 0.0001	**0.0064** ± 0.0001
	w/Un-JHgRF-Net	0.801 ± 0.001	1.212 ± 0.017	1.326 ± 0.038		w/Un-JHgRF-Net	0.0047 ± 0.0001	0.1193 ± 0.0001	0.118 ± 0.0002

Table 2
The pointwise forecast errors on benchmark datasets at horizon@12.

Model	PeMSD3			PeMSD4			PeMSD7			PeMSD8			PeMSD7(M)		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
HA	31.58	52.39	33.78	38.03	59.24	27.88	45.12	65.64	24.51	34.86	59.24	27.88	4.59	8.63	14.35
ARIMA	35.41	47.59	33.78	33.73	48.80	24.18	38.17	59.27	19.46	31.09	44.32	22.73	7.27	13.20	15.38
VAR	23.65	38.26	24.51	24.54	38.61	17.24	50.22	75.63	32.22	19.19	29.81	13.10	4.25	7.61	10.28
FC-LSTM	21.33	35.11	23.33	26.77	40.65	18.23	29.98	45.94	13.20	23.09	35.17	14.99	4.16	7.51	10.10
TCN	19.32	33.55	19.93	23.22	37.26	15.59	32.72	42.23	14.26	22.72	35.79	14.03	4.36	7.20	9.71
TCN(w/o causal)	18.87	32.24	18.63	22.81	36.87	14.31	30.53	41.02	13.88	21.42	34.03	13.09	4.43	7.53	9.44
GRU-ED	19.12	32.85	19.31	23.68	39.27	16.44	27.66	43.49	12.20	22.00	36.22	13.33	4.78	9.05	12.66
DSANet	21.29	34.55	23.21	22.79	35.77	16.03	31.36	49.11	14.43	17.14	26.96	11.32	3.52	6.98	8.78
STGCN	17.55	30.42	17.34	21.16	34.89	13.83	25.33	39.34	11.21	17.50	27.09	11.29	3.86	6.79	10.06
DCRNN	17.99	30.31	18.34	21.22	33.44	14.17	25.22	38.61	11.82	16.82	26.36	10.92	3.83	7.18	9.81
GraphWaveNet	19.12	32.77	18.89	24.89	39.66	17.29	26.39	41.50	11.97	18.28	30.05	12.15	3.19	6.24	8.02
ASTGCN(r)	17.34	29.56	17.21	22.93	35.22	16.56	24.01	37.87	10.73	18.25	28.06	11.64	3.14	6.18	8.12
MSTGCN	19.54	31.93	23.86	23.96	37.21	14.33	29.00	43.73	14.30	19.00	29.15	12.38	3.54	6.14	9.00
STG2Seq	19.03	29.83	21.55	25.20	38.48	18.77	32.77	47.16	20.16	20.17	30.71	17.32	3.48	6.51	8.95
LSGCN	17.94	29.85	16.98	21.53	33.86	13.18	27.31	41.46	11.98	17.73	26.76	11.20	3.05	5.98	7.62
STSGCN	17.48	29.21	16.78	21.19	33.65	13.90	24.26	39.03	10.21	17.13	26.80	10.96	3.01	5.93	7.55
AGCRN	15.98	28.25	15.23	19.83	32.26	12.97	22.37	36.55	9.12	15.95	25.22	10.09	2.79	5.54	7.02
STFGNN	16.77	28.34	16.30	20.48	32.51	16.77	23.46	36.60	9.21	16.94	26.25	10.60	2.90	5.79	7.23
STGODE	16.50	27.84	16.69	20.84	32.82	13.77	22.59	37.54	10.14	16.81	25.97	10.62	2.97	5.66	7.36
Z-GCNETs	16.64	28.15	16.39	19.50	31.61	12.78	21.77	35.17	9.25	15.76	25.11	10.01	2.75	5.62	6.89
STG-NCDE	15.57	27.09	15.06	19.21	31.09	12.76	**20.53**	33.84	**8.80**	15.45	24.81	9.92	**2.68**	5.39	6.76
JHgRF-Net	**14.18**	**21.48**	**12.19**	19.23	**28.76**	**11.55**	22.19	**32.89**	9.61	**14.34**	**22.06**	**8.29**	2.90	**5.32**	**6.77**
w/Un-JHgRF-Net	14.25	21.34	12.19	20.36	30.21	12.35	23.33	34.98	10.11	14.85	22.84	8.48	2.82	5.01	2.82

squared error(RMSE), and mean absolute percentage error(MAPE) to provide an accurate estimate of the models performance. We reported the baseline model results from [18]. Our experimental findings indicate that the proposed models(**JHgRF-Net** and **w/Unc–Net**) consistently outperformed the baseline models, exhibiting lower forecast errors across the different benchmark datasets. On the PeMSD3, PeMSD4, PeMSD7, PeMSD8, and PeMSD7(M) datasets, the proposed model(**JHgRF-Net**) demonstrated significant improvement over the next-best baseline models, achieving a reduction of 20.71%, 7.49%, 2.81%, 11.08%, and 1.30% in the RMSE metric, respectively. Apart from pointwise forecasts, the **w/Unc-JHgRF-Net** model(which integrates **JHgRF-Net** with local uncertainty estimation) predicts time-varying uncertainty estimates of the multi-horizon forecasts. While it exhibits slightly lower performance than the **JHgRF-Net** model, it still outperforms several robust baselines found in the literature, as demonstrated by the reduced prediction error. Additionally, in Table 1 , we show the performance of **JHgRF-Net** and **w/Unc-JHgRF-Net**, and several baseline models on the MTSF task across multiple datasets: METR-LA, PEMS-BAY, Solar-energy, Electricity, Exchange-rate, Traffic, SWaT and WADI . The models were evaluated using various metrics, including MAE, RMSE, and MAPE, and corresponding forecast errors were reported for 3-, 6-, and 12-steps ahead forecast horizons. The proposed models, **JHgRF-Net** and **w/Unc-JHgRF-Net**, demonstrated superior performance compared to the baseline models, with significantly lower forecast errors observed on all the datasets. On the METR-LA, PEMS-BAY, Solar-energy, Electricity, Exchange-rate, Traffic, SWaT and WADI datasets, the proposed model(**JHgRF-Net**) shows superior performance over the next-best baseline models, achieving a reduction of 37.01%, 21.74%, 54.93%, 3.77%, 37.14%, 18.18%, 51.25% and 17.63% in the RMSE metric, respectively for the 6-step ahead forecast horizon. Our empirical findings validate the efficacy of the proposed neural forecasting architecture to capture the complex nonlinear spatio-temporal dynamics that are present in MTS data, leading to improved forecasting performance. Please refer to the appendix, for further details on the experimental methodology, ablation studies, and additional experimental results. The appendix includes a comprehensive analysis of the **JHgRF-Net** model's ability to handle missing data, as well as a more detailed description of the **w/Unc-JHgRF-Net** model's ability to estimate uncertainty. Additionally, the appendix offers comprehensive visualizations of model predictions with uncertainty estimates in comparison to the ground truth, along with additional information on brief overview of the baseline models.

7. Conclusion

Our proposed forecasting architecture accurately models the complex spatio-temporal dynamics within MTS data and achieves accurate multi-horizon forecasts compared to the several baselines. The experimental results obtained from real-world datasets demonstrate the effectiveness of our approach, as supported by improved forecast estimates and reliable uncertainty estimations. In the future, our focus will be on expanding the framework's capabilities to handle large-scale graph datasets, enabling its utilization for a wide range of applications, such as anomaly detection, missing data imputation, etc.

References

[1] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, in: IJCAI, 2019, pp. 1907–1913.

[2] L. Bai, L. Yao, C. Li, X. Wang, C. Wang, Adaptive graph convolutional recurrent network for traffic forecasting, in: NeurIPS, 2020.

[3] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, 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.

[4] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, in: IJCAI, 2018, pp. 3634–3640.

[5] Y. Chen, I. Segovia-Dominguez, B. Coskunuzer, Y. Gel, TAMP-s2GCNets: Coupling time-aware multipersistence knowledge representation with spatio-supra graph convolutional networks for time-series forecasting, in: International Conference on Learning Representations, 2022.

[6] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, in: ICLR (Poster), 2018.

[7] Y. N. Dauphin, A. Fan, M. Auli, D. Grangier, Language modeling with gated convolutional networks, in: International conference on machine learning, PMLR, 2017, pp. 933–941.

[8] E. Jang, S. Gu, B. Poole, Categorical reparameterization with gumbel-softmax, arXiv preprint arXiv:1611.01144 (2016).

[9] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078 (2014).

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[11] J. L. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, arXiv preprint arXiv:1607.06450 (2016).

[12] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: CVPR, 2016.

[13] J. Gao, B. Ribeiro, On the equivalence between temporal and static equivariant graph representations, in: International Conference on Machine Learning, PMLR, 2022, pp. 7052–7076.

[14] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, Z. Jia, Freeway performance measurement system: mining loop detector data, Transportation Research Record 1748 (2001) 96–102.

[15] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, arXiv preprint arXiv:1707.01926 (2017).

[16] A. P. Mathur, N. O. Tippenhauer, Swat: A water treatment testbed for research and training on ics security, in: 2016 international workshop on cyber-physical systems for smart water networks (CySWater), IEEE, 2016, pp. 31–36.

[17] C. M. Ahmed, V. R. Palleti, A. P. Mathur, Wadi: a water distribution testbed for research in the design of secure cyber physical systems, in: Proceedings of the 3rd international workshop on cyber-physical systems for smart water networks, 2017, pp. 25–28.

[18] J. Choi, H. Choi, J. Hwang, N. Park, Graph neural controlled differential equations for traffic forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, 2022, pp. 6367–6374.

[19] I. Marisca, A. Cini, C. Alippi, Learning to reconstruct missing data from spatiotemporal graphs with sparse observations, arXiv preprint arXiv:2205.13479 (2022).

[20] A. Cini, D. Zambon, C. Alippi, Sparse graph learning for spatiotemporal time series, arXiv preprint arXiv:2205.13492 (2022).

[21] D. A. Nix, A. S. Weigend, Estimating the mean and variance of the target probability distribution, in: Proceedings of 1994 ieee international conference on neural networks (ICNN'94), volume 1, IEEE, 1994, pp. 55–60.

[22] A. Deng, B. Hooi, Graph neural network-based anomaly detection in multivariate time series, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, 2021, pp. 4027–4035.

[23] J. D. Hamilton, Time series analysis, Princeton university press, 2020.

[24] S. Bai, J. Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, arXiv:1803.01271 (2018).

[25] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: NeurIPS, 2014, pp. 3104–3112.

[26] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, in: EMNLP, 2014.

[27] S. Huang, D. Wang, X. Wu, A. Tang, Dsanet: Dual self-attention network for multivariate time series forecasting, in: CIKM, 2019.

[28] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, in: ICLR, 2018.

[29] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, in: IJCAI, 2018. URL: https://doi.org/10.24963/ijcai.2018/505. doi:10.24963/ijcai.2018/505.

[30] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, in: IJCAI, 2019, pp. 1907–1913.

[31] S. Guo, Y. Lin, N. Feng, C. Song, H. Wan, Attention based spatial-temporal graph convolutional networks for traffic flow forecasting, in: AAAI, 2019. doi:10.1609/aaai.v33i01.3301922.

[32] L. Bai, L. Yao, S. S. Kanhere, X. Wang, Q. Z. Sheng, Stg2seq: Spatial-temporal graph to sequence model for multi-step passenger demand forecasting, in: IJCAI, 2019. URL: https://doi.org/10.24963/ijcai.2019/274. doi:10.24963/ijcai.2019/274.

[33] C. Song, Y. Lin, S. Guo, H. Wan, Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting, in: AAAI, 2020. doi:10.1609/aaai.v34i01.5438.

[34] R. Huang, C. Huang, Y. Liu, G. Dai, W. Kong, Lsgcn: Long short-term traffic prediction with graph convolutional networks., in: IJCAI, 2020, pp. 2355–2361.

[35] L. Bai, L. Yao, C. Li, X. Wang, C. Wang, Adaptive graph convolutional recurrent network for traffic forecasting, in: NeurIPS, volume 33, 2020, pp. 17804–17815.

[36] M. Li, Z. Zhu, Spatial-temporal fusion graph neural networks for traffic flow forecasting,

in: AAAI, 2021.

[37] Y. Chen, I. Segovia-Dominguez, Y. R. Gel, Z-gcnets: Time zigzags at graph convolutional networks for time series forecasting, in: ICML, 2021.

[38] Z. Fang, Q. Long, G. Song, K. Xie, Spatial-temporal graph ode networks for traffic flow forecasting, in: KDD, 2021.

[39] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, R. Zemel, Neural relational inference for interacting systems, in: International Conference on Machine Learning, PMLR, 2018, pp. 2688–2697.

8. APPENDIX

8.1. Ablation Study

The **JHgRF-Net** framework, serving as the baseline for our ablation study, seamlessly integrates both spatial and temporal inference components to model complex inter- and intra-time series correlations in interconnected sensor networks. Its spatial inference component comprises of two modules: Spatio-Temporal Hypergraph Convolutional Network(STHgCN) and Spatio-Temporal Transformer Network(STTN). In an extensive ablation study, we evaluate the impact of each component in the **JHgRF-Net** framework on the MTSF task. By selectively removing components, we can observe the impact of individual components on the overall framework performance, gaining valuable insight into their unique contributions towards the framework effectiveness. The study conducted a systematic elimination and creation of various ablated variants to identify critical components that enhance the framework performance. By comparing the impact of these components on the MTSF task against the baseline, valuable insights were gained into each component contribution to the overall framework performance. The ablation study led to an improved understanding of the relationship between the various ablated variants and the baseline, which resulted in a better understanding of the mechanisms that underlie their generalization performance. We present detailed information on each ablated variant created by systematically removing specific components, as follows:

- "w/o - Spatial": A variant of **JHgRF-Net** framework that excluded the spatial inference component, and its degraded performance highlights the significance of using STHgCN and STTN neural operators for effective modeling of inter-series correlations among multiple time series variables present in complex interconnected sensor networks.
- "w/o - Temporal": A variant of **JHgRF-Net** that excluded the temporal inference component, and its deteriorated performance highlighted the importance of incorporating the temporal inference component for effectively modeling the time-varying inter-series dependencies within multiple time series variables present in complex sensor network-based dynamical systems.
- "w/o - STHgCN": A variant of **JHgRF-Net** that excluded the STHgCN method, and its substandard performance shed light on the importance of attention-based hypergraph convolution operation for modeling the spatio-temporal dynamics present in the high-dimensional sensor network-based dynamic systems.
- "w/o - STTN": A variant of **JHgRF-Net** that excluded the STTN method, and its subpar performance emphasized the significance of hypergraph transformer networks, which utilizes full attention as a structural inductive bias for modeling the complex dynamics present in the high-dimensional interconnected sensor networks.

In Tables 3 - 9, we present the findings of our ablation studies on benchmark datasets. We employed multiple forecasting accuracy metrics, including Mean Absolute Error(MAE), Root Mean Squared Error(RMSE), and Mean Absolute Percentage Error(MAPE), to offer a comprehensive understanding of the relative performance of ablated variants compared to the baseline. We evaluated the accuracy of multistep-ahead forecasting task by comparing pointwise forecasts with observed data(ground-truth) during the prediction interval and the results were reported

using the previously mentioned forecast accuracy metrics. For additional clarity, we enclosed the relative percentage difference between the ablated variants and the baseline performance within parentheses. To ensure the accuracy of our findings, we conducted multiple experiments and reported the average results. Moreover, we evaluated the ablated variants ability to handle long-term predictions by setting the forecast horizon to 12 and comparing it with the baseline. Tables 3 - 9, demonstrate that the ablated variants have lower forecast accuracy and perform considerably worse than the baseline. Upon closer examination, it is apparent that, for achieving state-of-the-art performance on benchmark datasets, the spatial inference component within the **JHgRF-Net** framework is more important than the temporal inference component. The ablation studies yielded the following observations:

- On the PeMSD8 dataset, analysis indicates that the "w/o - Spatial" variant shows a significant decline in performance relative to the baseline, with an increase of 21.12% in RMSE, 27.55% in MAE, and 40.77% in MAPE. Conversely, the "w/o - Temporal" variant exhibits slightly inferior performance compared to the baseline, with a modest rise of 16.23% in RMSE, 14.02% in MAE, and 10.15% in MAPE.

- Likewise, similar trends are observed on the PeMSD4 dataset. The "w/o - Spatial" variant significantly underperforms the benchmark, with an increase of 20.86% in RMSE, 22.26% in MAE, and 22.34% in MAPE. In contrast, the "w/o - Temporal" variant exhibits a minor reduction in its performance when compared to the baseline, with a marginal rise of 8.76% in RMSE, 4.47% in MAE, and 2.86% in MAPE.

- Analogous trends are observed for PeMSD7 dataset. In particular, the "w/o - Spatial" variant displays a notable decline in performance relative to the baseline, with an increase of 18.33% in RMSE, 21.05% in MAE, and 44.33% in MAPE. On the other hand, the "w/o - Temporal" variant indicates a minor drop in performance compared to the baseline, with a slight increase of 8.63% in RMSE, 7.75% in MAE, and 8.74% in MAPE.

The higher increase in the error metrics of the ablated variants performance, in comparison to the baseline, further emphasizes the relative significance of the mechanisms underlying the excluded components of the baseline. To put it briefly, the spatial inference component serves as a powerful backbone that fortifies the **JHgRF-Net** framework for improving forecasting performance. This component is responsible for capturing the intricate dependencies among multiple time series variables and learning the dynamics of interacting systems. The crucial role of the spatial inference component is evident from the substantial decline in performance when it is excluded as compared to the baseline, emphasizing its indispensable nature. Our proposed neural forecast architecture is built upon two fundamental methods known as the STHgCN and STTN neural operators, which collectively make up the spatial inference component. The following observations were made from the ablation studies:

- The "w/o - STHgCN" variant yielded inferior results compared to the benchmark, with a difference of 12.03%, 10.66%, and 11.43% in terms of RMSE, MAE, and MAPE metrics, respectively, for PeMSD4 dataset. Similarly on PeMSD8, the variants exhibited a 10.52%, 9.97%, and 9.89% decrease in performance with respect to the same metrics as compared to the benchmark. These results provide evidence in support of the notion that incorporating

STHgCN method in the learning process can result in better performance in multi-horizon forecasting tasks.

- The "w/o - STTN" variant exhibited a slight increase in the RMSE, MAE, and MAPE metrics compared to the baseline, with differences of 3.79%, 0.31%, and 0.61% on PeMSD4, and 1.90%, 1.88%, and 2.29% on PeMSD8, respectively. Nonetheless, the integration of the STTN method was found to be crucial, as it resulted in a notable improvement in forecast accuracy.

Table 3

The table presents the results of an ablation study on multi-horizon forecasting using the PeMSD3 and PeMSD4 benchmark datasets.

Method		MAE	RMSE	MAPE		MAE	RMSE	MAPE
JHgRF-Net	PeMSD3	**14.18**	**21.48**	**12.19**	PeMSD4	**19.23**	**28.76**	**11.55**
w/o - Spatial		17.84(20.52%↑)	25.68(19.55%↑)	15.65(28.38%↑)		23.51(22.26%↑)	34.76(20.86%↑)	14.13(22.34%↑)
w/o - Temporal		15.07(6.28%↑)	23.91(11.31%↑)	13.04(6.97%↑)		20.09(4.47%↑)	31.28(8.76%↑)	11.88(2.86%↑)
w/o - STHgCN		16.55(16.71%↑)	25.64(19.37%↑)	13.99(14.77%↑)		21.28(10.66%↑)	32.22(12.03%↑)	12.87(11.43%↑)
w/o - STTN		14.27(0.63%↑)	21.84(1.68%↑)	12.34(1.23%↑)		19.29(0.31%↑)	29.85(3.79%↑)	11.62(0.61%↑)

Table 4

The table presents the results of an ablation study on multi-horizon forecasting using the PeMSD7 and PeMSD8 benchmark datasets.

Method		MAE	RMSE	MAPE		MAE	RMSE	MAPE
JHgRF-Net	PeMSD7	**22.19**	**32.89**	**9.61**	PeMSD8	**14.34**	**22.06**	**8.29**
w/o - Spatial		26.86(21.05%↑)	38.92(18.33%↑)	13.87(44.33%↑)		18.29(27.55%↑)	26.72(21.12%↑)	11.67(40.77%↑)
w/o - Temporal		23.91(7.75%↑)	35.73(8.63%↑)	10.45(8.74%↑)		16.35(14.02%↑)	25.64(16.23%↑)	10.15(22.44%↑)
w/o - STHgCN		25.11(13.16%↑)	36.66(11.46%↑)	9.13(4.99%↓)		15.77(9.97%↑)	24.38(10.52%↑)	9.11(9.89%↑)
w/o - STTN		22.89(3.15%↑)	38.32(16.51%↑)	10.95(13.94%↑)		14.61(1.88%↑)	22.48(1.90%↑)	8.48(2.29%↑)

Table 5

The table presents the results of an ablation study on multi-horizon forecasting using the PeMSD7(M) benchmark dataset.

Method		MAE	RMSE	MAPE
JHgRF-Net	PeMSD7(M)	**2.90**	**5.32**	**6.77**
w/o - Spatial		3.87(33.45%↑)	6.78(27.44%↑)	7.75(14.48%↑)
w/o - Temporal		3.06(5.52%↑)	5.83(9.59%↑)	6.92(2.22%↑)
w/o - STHgCN		2.95(1.72%↑)	5.38(1.13%↑)	6.82(0.74%↑)
w/o - STTN		3.02(4.14%↑)	5.53(3.95%↑)	6.95(2.66%↑)

Table 6

The table presents the results of an ablation study on multi-horizon forecasting using the METR-LA and SWaT benchmark datasets.

Method		MAE				MAE		
		Horizon @ 3	Horizon @ 6	Horizon @ 12		Horizon @ 3	Horizon @ 6	Horizon @ 12
JHgRF-Net	METR-LA	**2.039** ± 0.010	**2.059** ± 0.02	**4.937** ± 0.015	SWaT	**0.074** ± 0.006	**0.136** ± 0.012	**0.170** ± 0.003
w/o - Spatial		3.17(55.47%↑)	3.56(72.90%↑)	6.03(22.14%↑)		0.623(741.90%↑)	0.585(330.15%↑)	0.698(310.59%↑)
w/o - Temporal		2.33(14.27%↑)	2.94(42.79%↑)	4.65(5.81%↓)		0.325(339.19%↑)	0.334(145.59%↑)	0.405(138.24%↑)
w/o - STHgCN		2.185(7.16%↑)	3.412(65.71%↑)	5.101(3.32%↑)		0.481(550%↑)	0.475(249.26%↑)	0.519(205.29%↑)
w/o - STTN		2.051(0.59%↑)	3.267(58.67%↑)	4.957(0.40%↑)		0.102(37.84%↑)	0.143(5.15%↑)	0.206(21.18%↑)

Based on the ablation studies, we can conclude that STHgCN method is more effective than the STTN method in accurately modeling spatio-temporal dependencies in MTS data, leading to

better multi-horizon forecasts. Additional results from the ablation study on benchmark datasets are presented in Tables 3 - 9. The results indicate that the proposed **JHgRF-Net** framework exhibits strong generalization capabilities, even when dealing with intricate patterns across an extensive variety of datasets, and it can efficiently scale to handle large-scale graph datasets. In summary, the ablation studies provide evidence in favor of the hypothesis that joint optimization of spatial-temporal inference components can lead to enhanced performance in multi-horizon forecasting tasks. In addition, the experimental findings support the rationale of inclusion of STHgCN and STTN neural operators to model the interdependencies among the multiple variables and learn the dynamics of the complex interconnected systems.

Table 7

The table presents the results of an ablation study on multi-horizon forecasting using the PeMS-BAY and Traffic benchmark datasets.

Method		MAE				MAE		
		Horizon @ 3	Horizon @ 6	Horizon @ 12		Horizon @ 3	Horizon @ 6	Horizon @ 12
JHgRF-Net	PeMS-BAY	0.806	1.217	1.758	Traffic	0.006	0.009	0.011
w/o - Spatial		1.17(45.16%↑)	1.465(20.38%↑)	2.095(19.17%↑)		0.0198(230%↑)	0.0163(81.11%↑)	0.0173(57.27%↑)
w/o - Temporal		0.976(21.09%↑)	1.341(10.19%↑)	1.933(9.95%↑)		0.0073(21.67%↑)	0.0142(57.78%↑)	0.0162(47.27%↑)
w/o - STHgCN		0.863(7.07%↑)	1.272(4.52%↑)	1.831(4.15%↑)		0.0144(140%↑)	0.0139(54.44%↑)	0.0151(37.27%↑)
w/o - STTN		0.814(0.99%↑)	1.251(2.79%↑)	1.769(0.63%↑)		0.0067(11.67%↑)	0.0101(12.22%↑)	0.0116(5.45%↑)

Table 8

The table presents the results of an ablation study on multi-horizon forecasting using the WADI and Electricity benchmark datasets.

Method		MAE				MAE		
		Horizon @ 3	Horizon @ 6	Horizon @ 12		Horizon @ 3	Horizon @ 6	Horizon @ 12
JHgRF-Net	WADI	4.149	4.592	4.758	Electricity	160.955	238.781	225.724
w/o - Spatial		6.031(45.36%↑)	5.982(30.27%↑)	5.733(20.49%↑)		209.325(30.05%↑)	312.475(30.86%↑)	298.685(32.32%↑)
w/o - Temporal		4.865(17.26%↑)	4.803(4.59%↑)	4.843(1.79%↑)		176.089(9.40%↑)	263.168(10.21%↑)	247.905(9.83%↑)
w/o - STHgCN		5.171(24.63%↑)	5.003(8.95%↑)	4.917(3.34%↑)		183.040(13.72%↑)	269.294(12.78%↑)	265.346(17.55%↑)
w/o - STTN		4.197(1.16%↑)	4.744(3.31%↑)	4.835(1.61%↑)		168.228(4.52%↑)	257.872(7.99%↑)	238.459(5.64%↑)

Table 9

The table presents the results of an ablation study on multi-horizon forecasting using the Solar Energy and Exchange Rate benchmark datasets.

Method		MAE				MAE		
		Horizon @ 3	Horizon @ 6	Horizon @ 12		Horizon @ 3	Horizon @ 6	Horizon @ 12
JHgRF-Net	Solar Energy	0.575 ± 0.013	0.868	0.873	Exchange Rate	0.0043	0.0044	0.0064
w/o - Spatial		1.132(96.87%↑)	1.059(22%↑)	3.087(253.61%↑)		0.0284(560.47%↑)	0.0182(313.64%↑)	0.0178(178.12%↑)
w/o - Temporal		0.726(26.26%↑)	0.883(1.73%↑)	1.898(117.41%↑)		0.0097(125.58%↑)	0.0081(84.09%↑)	0.0091(42.18%↑)
w/o - STHgCN		0.826(43.65%↑)	0.896(3.23%↑)	2.132(144.22%↑)		0.0122(183.72%↑)	0.0085(93.18%↑)	0.0093(45.31%↑)
w/o - STTN		0.596(3.65%↑)	0.874(0.69%↑)	1.412(61.74%↑)		0.0066(53.49%↑)	0.0076(72.73%↑)	0.0086(34.37%↑)

8.2. Prediction error for multi-horizon forecasting

We conducted comprehensive experiments to evaluate the capability of the neural forecasting architecture, **JHgRF-Net**, to generate accurate multi-horizon forecasts on several benchmark datasets. The forecast errors of the **JHgRF-Net** framework performance on benchmark datasets are shown in Figure 10. The framework performance was evaluated using various metrics, such as MAPE and MAE. Lower values of forecast errors indicate better model performance. The results demonstrate that the framework outperformed the baselines on all the prediction horizons.

Table 10

The figure shows the pointwise prediction error for multi-horizon forecasting tasks on benchmark datasets.
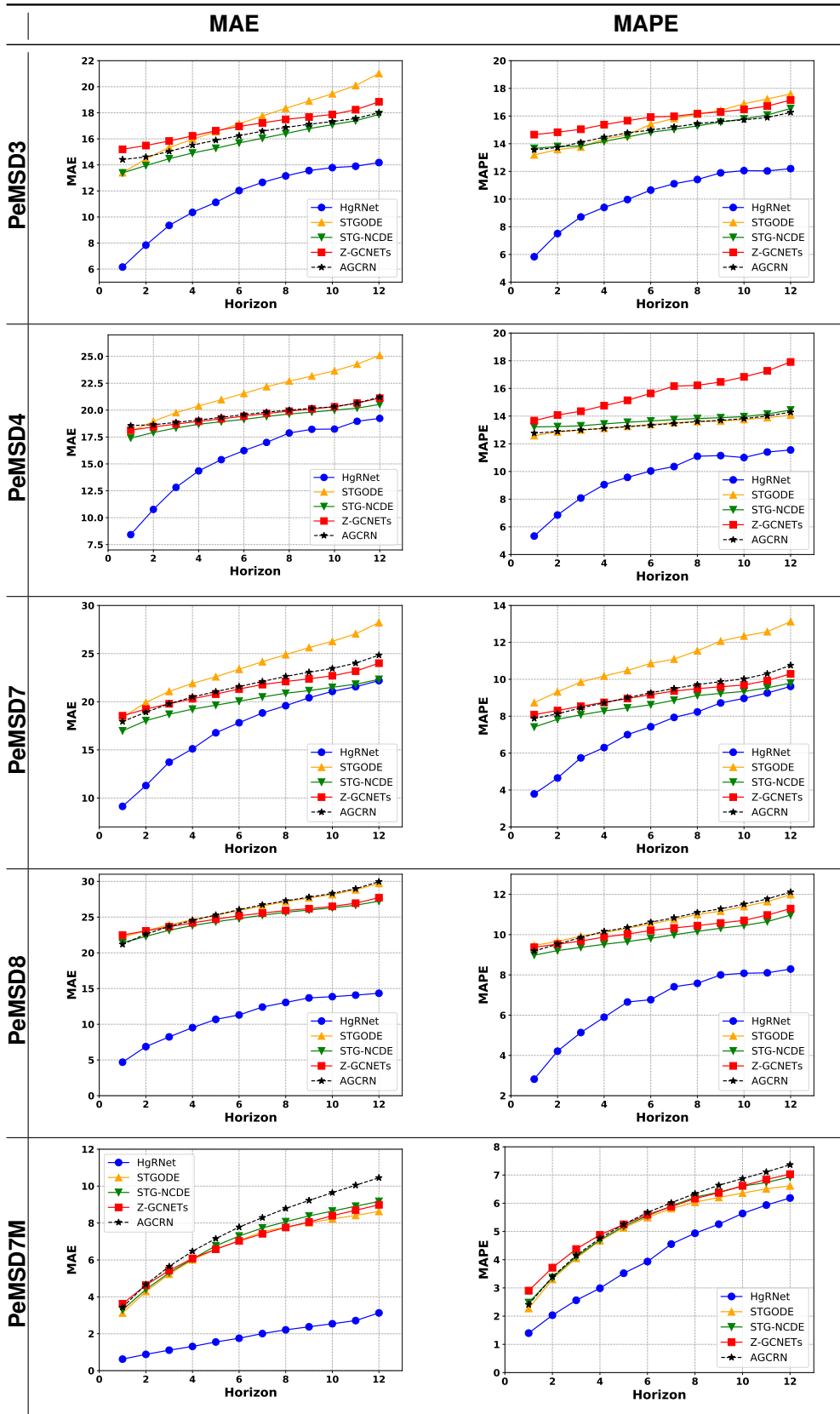
These findings suggest that the proposed framework has the potential to accurately model the nonlinear spatio-temporal dependencies and improve multi-horizon forecast accuracy through effectively exploiting the relational inductive biases within the hypergraph-structured MTS data.

8.3. Irregular time series forecasting

The **JHgRF-Net** framework ability to handle missing data in large, complex sensor networks was evaluated by simulating two commonly observed missingness patterns([19], [20]): point-missing and block-missing patterns. These patterns were created to mimic the missingness patterns observed in real-world data of such complex interconnected sensor networks. In point-missing pattern, observations of each variable were randomly dropped within a historical window, with missing ratios of 10%, 30%, and 50%. Similarly, in block-missing pattern, available data for each variable was randomly masked within a historical window, also with missing ratios ranging from 10%, 30%, and 50%. Moreover, sensor failures were simulated with a probability of 0.15%, leading to blocks of missing data for the multivariate time series data. To evaluate the **JHgRF-Net** framework performance on MTS data with missing values and to analyze the impact of increasing missing data percentage on framework performance, we split several benchmark datasets into three mutually exclusive sets - training, validation, and test - based on their chronological order. The METR-LA and PEMS-BAY datasets were split in a ratio of 7:1:2, while the other datasets(PeMSD3, PeMSD4, PeMSD7, PeMSD8, and PeMSD7(M), Traffic, Solar-Energy, Electricity, Exchange-Rate) were split in a ratio of 6:2:2. We utilized multiple forecasting metrics to evaluate the **JHgRF-Net** framework performance in handling missing data. We trained the **JHgRF-Net** framework on fully observed data to establish a benchmark for the MTSF task with missing values. The tables 11 - 15 present the forecasting results of the framework performance on the irregular-time-series datasets. The experimental studies demonstrate that the **JHgRF-Net** framework is reliable and robust in handling missing data, which is widely prevalent in real-world applications. The framework performance deteriorates slightly compared to the benchmark when there is a lower percentage of missing data. With a further increase in the percentage of missing data, the framework performance continues to decline, resulting in lower forecast accuracy across all benchmark datasets, regardless of the missing data pattern. Instead of relying on imputed values for model predictions, the proposed framework utilizes observed data for multi-horizon forecasting, hence demonstrates its robustness to handle missing data. Moreover, by capturing complex dependencies and patterns within multivariate time series data present in interconnected networks, the framework generates more dependable out-of-sample forecasts, resulting in enhanced multi-horizon forecast accuracy.

8.4. Sensitivity analysis

We carried out a hyperparameter study to evaluate the impact of specific hyperparameters on the proposed framework performance. Our aim was to find the ideal set of hyperparameter values that could result in achieving the best possible performance on the benchmark datasets. We tuned four hyperparameters - embedding size(d), number of hyperedges($|\mathbf{E}|$), batch size(b), and learning rate(lr) - within specific ranges of values. The opted ranges were as follows: $d \in \{2, 6, 10, 18, 24\}$, $|\mathbf{E}| \in \{2, 5, 8\}$, $b \in \{2, 6, 10, 18, 24, 32, 64\}$, and $lr \in \{1 \times 10^{-1}, 1 \times 10^{-2}, 1 \times 10^{-3}, 1 \times 10^{-4}\}$. We have carefully selected ranges for the hyperparameters to prevent memory errors and limit

Table 11
Pointwise forecasting error on irregular PeMSD3, PeMSD4 and PeMSD7 datasets at horizon@12

Missing Rate	Model		MAE	RMSE	MAPE		MAE	RMSE	MAPE		MAE	RMSE	MAPE
0%	JHgRF-Net		14.18	21.48	12.19		19.23	28.76	11.55		22.19	32.89	9.61
10%	w/Point	PeMSD3	15.09	22.79	12.88	PeMSD4	20.91	30.63	12.87	PeMSD7	23.77	34.83	10.23
	w/Block		15.04	22.74	12.77		20.87	30.64	30.64		23.61	34.70	10.17
30%	w/Point		15.58	23.45	13.16		21.79	31.73	13.27		24.99	36.32	10.81
	w/Block		15.60	23.49	13.13		21.63	31.61	13.19		24.98	36.31	10.84
50%	w/Point		16.34	24.42	13.84		23.12	33.49	14.11		26.65	38.47	11.69
	w/Block		16.20	24.29	13.66		23.24	33.84	14.11		26.63	38.33	11.54

Table 12
Pointwise forecasting error on irregular PeMSD7(M) and PeMSD8 datasets at horizon@12

Missing Rate	Model		MAE	RMSE	MAPE		MAE	RMSE	MAPE
0%	JHgRF-Net		2.83	5.01	6.18		14.34	22.06	8.29
10%	w/Point	PeMSD7(M)	3.34	5.53	7.14	PeMSD8	16.95	25.23	9.94
	w/Block		3.27	5.47	7.02		16.85	25.08	9.84
30%	w/Point		3.56	5.78	7.54		17.48	25.89	10.14
	w/Block		3.48	5.72	7.42		17.51	25.95	10.17
50%	w/Point		3.66	5.93	7.71		18.72	27.53	10.88
	w/Block		3.76	6.02	7.89		18.58	27.36	10.88

Table 13
Pointwise forecasting error on irregular METR-LA, PeMS-BAY and SWaT datasets in terms of MAE

Missing Rate	Model		MAE				MAE				MAE		
			Horizon@3	Horizon@6	Horizon@12		Horizon@3	Horizon@6	Horizon@12		Horizon@3	Horizon@6	Horizon@12
0%	JHgRF-Net		2.039	2.059	4.937		0.806	1.217	1.758		0.074	0.136	0.17
10%	w/Point	METR-LA	2.083	2.093	4.951	PeMS-BAY	0.813	1.225	1.763	SWaT	0.076	0.138	0.174
	w/Block		2.127	2.138	4.983		0.824	1.239	1.767		0.078	0.141	0.178
30%	w/Point		2.137	2.143	5.113		0.842	1.241	1.772		0.086	0.153	0.188
	w/Block		2.175	2.189	5.267		0.856	1.253	1.793		0.091	0.164	0.198
50%	w/Point		2.166	2.178	5.671		0.861	1.379	1.845		0.099	0.189	0.201
	w/Block		2.181	2.193	5.703		0.878	1.384	1.863		0.112	0.197	0.211

Table 14
Pointwise forecasting error on irregular Electricity, Exchange Rate and Solar Energy datasets in terms of MAE

Missing Rate	Model		MAE				MAE				MAE		
			Horizon@3	Horizon@6	Horizon@12		Horizon@3	Horizon@6	Horizon@12		Horizon@3	Horizon@6	Horizon@12
0%	JHgRF-Net		0.575	0.868	0.873		160.955	178.781	225.724		0.0043	0.0044	0.0064
10%	w/Point	Solar Energy	0.579	0.871	0.877	Electricity	163.072	181.213	228.358	Exchange Rate	0.00435	0.00456	0.00653
	w/Block		0.588	0.879	0.893		165.321	184.369	230.967		0.00437	0.00459	0.0066
30%	w/Point		0.613	0.911	0.937		170.325	192.565	244.637		0.00553	0.00578	0.00641
	w/Block		0.637	0.943	0.981		173.368	200.098	257.974		0.00693	0.00702	0.00725
50%	w/Point		0.689	0.988	1.103		189.765	215.131	274.387		0.00778	0.00869	0.00913
	w/Block		0.713	0.997	1.119		198.508	221.965	293.365		0.00896	0.00913	0.0132

Table 15

Pointwise forecasting error on irregular WADI and Traffic datasetsin terms of MAE

Missing Rate	Model		MAE				MAE		
			Horizon@3	Horizon@6	Horizon@12		Horizon@3	Horizon@6	Horizon@12
0%	JHgRF-Net		4.149	4.592	4.758		0.006	0.009	0.011
10%	w/Point	WADI	4.153	4.602	4.761	Traffic	0.0063	0.0092	0.0113
	w/Block		4.155	4.608	4.773		0.0065	0.0095	0.0117
30%	w/Point		4.236	4.703	4.893		0.00723	0.0103	0.0123
	w/Block		4.264	4.796	4.913		0.00768	0.0118	0.0133
50%	w/Point		4.286	4.811	5.031		0.00898	0.0163	0.0138
	w/Block		4.293	4.977	5.113		0.00955	0.0171	0.0145

model size. We optimized the framework hyperparameters through grid search and measured the model performance by measuring metrics like MAE and RMSE. These experimental results offered valuable insights into the effect of these hyperparameters on the framework ability to produce accurate forecasts in multivariate time series analysis, enhancing our understanding of its overall performance. The optimal hyperparameter configurations that yielded the best performance for each dataset are presented below,

- For PeMSD3, we set the batch size(b) to 18, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. Additionally, the number of hyperedges is 5.
- For PeMSD4, we set the batch size(b) to 32, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. Additionally, the number of hyperedges is 5.
- For PeMSD7, we set the batch size(b) to 6, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. Additionally, the number of hyperedges is 6.
- For PeMSD8, we set the batch size(b) to 48, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. Additionally, the number of hyperedges is 8.
- For PeMSD7(M), we set the batch size(b) to 48, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. The number of hyperedges is 6.
- For METR-LA, we set the batch size(b) to 48, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. Additionally, the number of hyperedges is 5.
- For PEMS-BAY, we set the batch size(b) to 12, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. The number of hyperedges is 5.
- For SWAT, we set the batch size(b) to 256, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. Additionally, the number of hyperedges is 5.
- For WADI, we set the batch size(b) to 64, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 12. Additionally, the number of hyperedges is 5.
- For Electricity, we set the batch size(b) to 32, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. Additionally, the number of hyperedges is 2.
- For Solar-energy, we set the batch size(b) to 32, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. The number of hyperedges is 6.
- For Exchange-rate, we set the batch size(b) to 32, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. The number of hyperedges is 6.
- For Traffic, we set the batch size(b) to 8, the initial learning rate(lr) to 1×10^{-3}, and the embedding size(d) to 18. Additionally, the number of hyperedges is 5.

The proportion of hypernodes connected to hyperedges in a hypergraph indicates the network's "edge density". A higher fraction of connected hypernodes suggests a denser network, while a lower fraction implies a sparser network. Modifying the number of hyperedges enables control over the hypergraph's density. The hyperparameter study yields the optimal number of hyperedges for an MTSF task by evaluating the impact of the number of predefined hyperedges on the learned hypergraph structures. This study sheds light on how the hypergraph's density changes as the number of hyperedges increases or decreases for a particular dataset in the MTSF task, with Table 16 presenting the experimental results.

Table 16
Experimental results of the hyperparameter study on the benchmark datasets.

Hyperparameter		METR-LA RMSE	METR-LA MAE	Solar Energy RMSE	Solar Energy MAE	SWaT RMSE	SWaT MAE	WADI RMSE	WADI MAE	Traffic RMSE	Traffic MAE
Embedded Dimension	2	9.4477	5.56	3.6908	2.4007	1.8902	0.435	54.8259	88.1317	0.0276	0.0276
	6	9.0538	5.1174	2.5359	1.5693	0.8783	0.2213	43.879	5.0819	0.0207	0.0121
	12	8.9438	5.1621	2.3676	1.4146	0.7175	0.1839	43.0202	4.691	0.0197	0.0114
	18	8.8559	5.0357	2.3268	1.3956	0.7087	0.168	42.9041	4.766	0.0194	0.0112
Number of Hyperedges	2	8.8461	5.0311	2.3598	1.4037	0.7733	0.1897	43.4547	5.0108	0.0193	0.0111
	5	8.8559	5.0365	2.3261	1.3946	0.7177	0.1839	43.112	4.7899	0.0194	0.0111
	6	8.8487	5.0837	2.2706	1.3412	0.7764	0.1907	43.9438	5.1023	0.0194	0.0111
	8	8.8794	5.0472	2.3323	1.3982	0.7594	0.7594	42.8957	4.7274	0.0195	0.0112

Hyperparameter		PeMSD3 RMSE	PeMSD3 MAE	PeMSD4 RMSE	PeMSD4 MAE	PeMSD7 RMSE	PeMSD7 MAE	PeMSD8 RMSE	PeMSD8 MAE	PeMSD7(M) RMSE	PeMSD7(M) MAE
Embedded Dimension	2	29.8281	19.3176	41.3629	29.3875	39.879	26.6195	29.7079	20.4021	6.1256	3.8583
	6	29.938	15.9012	32.521	21.7021	32.8907	22.1893	25.7841	16.9532	5.492	3.1196
	12	22.1569	14.4857	29.7458	19.9336	31.9132	21.4281	23.658	15.3998	5.2024	2.9499
	18	21.5551	14.2369	28.7556	19.2327	30.9304	20.6119	22.6019	14.6955	5.0874	2.9073
Number of Hyperedges	2	21.5534	14.3112	29.1418	19.4152	31.4684	21.0394	23.042	15.0624	5.1011	2.9132
	5	21.5748	14.243	28.768	19.2375	31.0854	20.7662	22.6021	14.6956	5.0866	2.9067
	6	21.646	14.2859	29.1065	19.4	31.0845	20.7209	22.558	14.6561	5.0333	2.8265
	8	21.8008	14.3245	28.6527	19.1951	31.7462	21.2531	22.3803	14.5223	5.0403	2.8618

Hyperparameter		Exchange Rate RMSE	Exchange Rate MAE	Electricity RMSE	Electricity MAE	PeMS-BAY RMSE	PeMS-BAY MAE
Embedded Dimension	2	0.037	0.0267	2806.454	384.8923	3.5032	1.8714
	6	0.0136	0.0091	1504.247	252.516	3.2833	1.8136
	12	0.0128	0.0085	1323.743	232.582	3.2317	1.7753
	18	0.0128	0.0084	1304.031	228.0085	3.2071	1.7585
Number of Hyperedges	2	0.0125	0.0082	1222.109	222.5563	3.2255	1.7822
	5	0.0128	0.0084	1303.631	227.898	3.2077	1.7588
	6	0.0124	0.0081	1275.573	224.6772	3.2198	1.7646
	8	0.0127	0.0083	1262.914	225.6226	3.2058	1.7533

8.5. Time series forecasting visualization

Figure 3 depicts the ground truth, pointwise forecasts, and time-varying uncertainty estimates obtained from the proposed **w/Unc-JHgRF-Net** framework. The visualizations provide valuable insights into the framework performance and facilitates comprehensive analysis and result interpretation. Existing methods for MTSF can model nonlinear spatio-temporal dependencies within interconnected sensor networks but often fail to provide accurate measures of uncertainty. In contrast, the proposed **w/Unc-JHgRF-Net** framework(**JHgRF-Net** framework with local uncertainty estimation) effectively utilizes relational inductive bias via spatio-temporal propagation architecture to quantitatively estimate uncertainty of multi-horizon forecasts. The framework accurately estimates uncertainty, outperforming existing methods that solely provide pointwise forecasts for MTSF. The multifaceted visualizations shows the framework effectiveness in time series representation learning for the MTSF task, making it a valuable contribution to the field of multivariate time series analysis for uncertainty estimation.
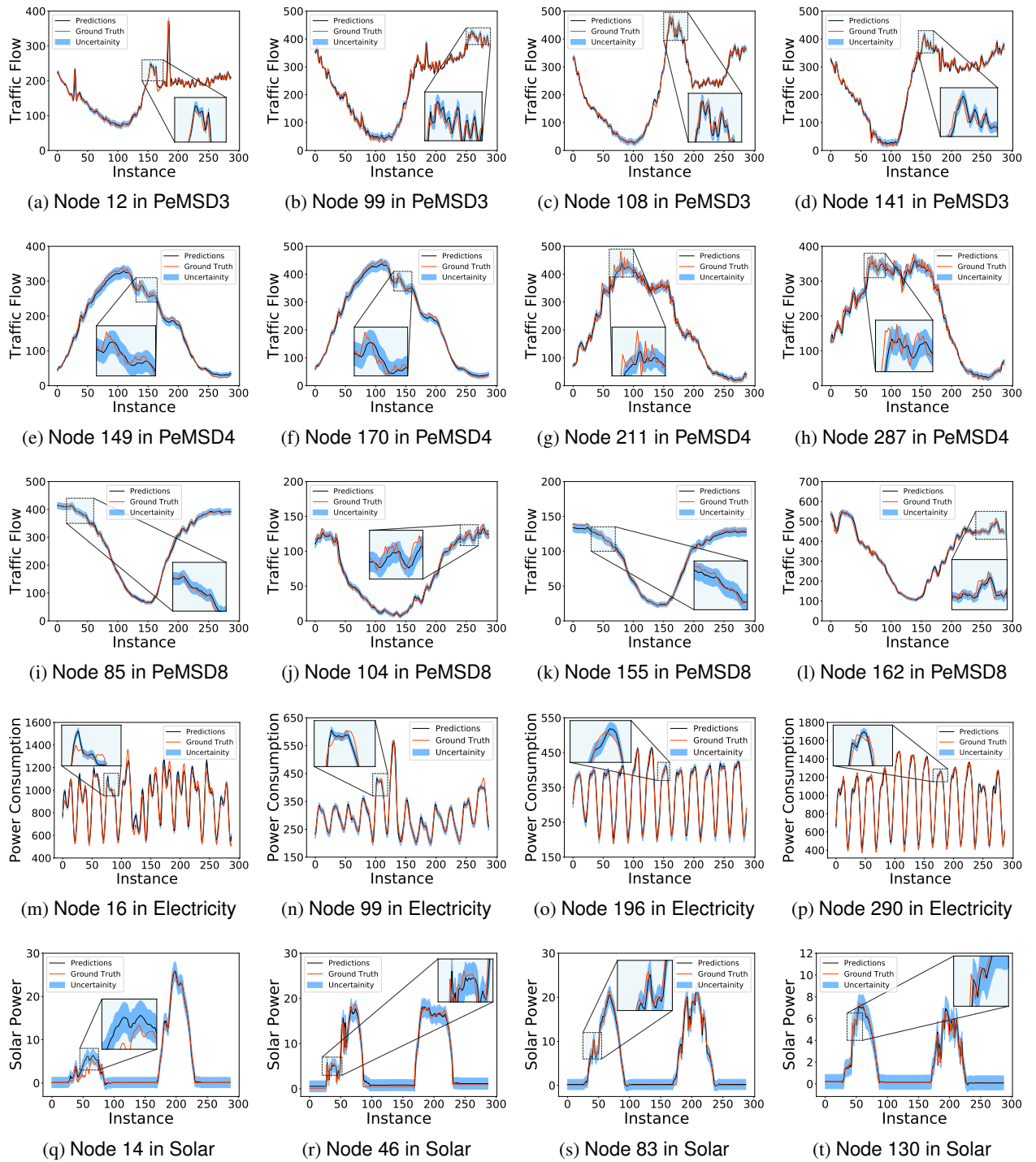
Figure 3: The visualizations provide an overview of the model forecasts, ground-truth data, and uncertainty estimates obtained through multi-horizon forecasting on benchmark datasets.

8.6. Forecasting uncertainty

The loss function for training the **JHgRF-Net** framework is the mean absolute error(MAE), which is calculated by comparing the pointwise forecasts of the model predictions ($\widehat{\mathbf{X}}_{(t:t+v-1)}$) with the corresponding ground-truth data($\mathbf{X}_{(t:t+v-1)}$), computed as follows:

$$\mathcal{L}_{\text{MAE}}(\theta) = \frac{1}{v}\left|\mathbf{X}_{(t:t+v-1)} - \widehat{\mathbf{X}}_{(t:t+v-1)}\right|$$

During the training process, the model parameters θ are fine-tuned to minimize the mean absolute error(MAE) loss function, represented as $\mathcal{L}_{\text{MAE}}(\theta)$. The **w/Unc-JHgRF-Net** is a variant of the **JHgRF-Net** that estimates the uncertainty in model predictions to enhance the reliability of decision-making. The framework utilizes a heteroscedastic Gaussian distribution to predict time-varying uncertainty in model predictions, characterized by mean and variance denoted by $\mu_\phi(\overline{\mathbf{X}}_{(t:t+v-1)})$ and $\sigma_\phi^2(\overline{\mathbf{X}}_{(t:t+v-1)})$, respectively, while the input time series is denoted as $\overline{\mathbf{X}}_{(t:t+v-1)}$. It is mathematically described as follows,

$$\widehat{\mathbf{X}}_{(t:t+v-1)} \sim \mathcal{N}\left(\mu_\phi(\overline{\mathbf{X}}_{(t:t+v-1)}), \sigma_\phi^2(\overline{\mathbf{X}}_{(t:t+v-1)})\right)$$

The predicted mean and standard deviation can be obtained from the following equation, as follows:

$$\mu_\phi(\overline{\mathbf{X}}_{(t:t+v-1)}), \sigma_\phi^2(\widehat{\mathbf{X}}_{(t:t+v-1)}) = f_\theta(\widehat{\mathbf{X}}_{(t:t+v-1)})$$

A neural network f_θ takes the output of the spatio-temporal inference component, represented by $\widehat{\mathbf{X}}_{(t:t+v-1)}$, and predicts the mean and standard deviation of future observations. The future observations are represented by $\widehat{\mathbf{X}}_{(t:t+v-1)}$ and it is the Maximum likelihood estimation(MLE) of the predicted Gaussian distribution. It is mathematically described as follows,

$$\widehat{\mathbf{X}}_{(t:t+v-1)} = \mu_\phi(\overline{\mathbf{X}}_{(t:t+v-1)})$$

To predict a Gaussian(normal) distribution for sampling future observations, we typically use the maximum likelihood estimates(MLE) of the distribution's parameters, namely the mean and standard deviation. The MLE values are obtained by maximizing the likelihood of the observed data being generated by the predicted distribution. In simpler terms, $\mu_\phi(\overline{\mathbf{X}}_{(t:t+v-1)})$ estimates the future values, $\widehat{\mathbf{X}}_{(t:t+v-1)}$, using the input time series, $\overline{\mathbf{X}}_{(t:t+v-1)}$. Meanwhile, $\sigma_\phi^2(\overline{\mathbf{X}}_{(t:t+v-1)})$ predicts the uncertainty in the model's predictions over the next v time steps, starting from the current time point, t. The uncertainty modeling framework optimizes the negative Gaussian log likelihood([21]) of the observations, based on estimates of the mean and variance. This approach provides a more comprehensive understanding and measurement of prediction uncertainty. The negative Gaussian log likelihood measures the likelihood of the observations, given the estimated mean and variance of the Gaussian distribution. A lower negative Gaussian log likelihood indicates a better fit of the Gaussian distribution to the observed values. The Gaussian distribution to predict the future observations is described by,

$$\mathcal{N}(\widehat{\mathbf{X}}_{(t:t+v-1)}; \mu_\phi(\overline{\mathbf{X}}_{(t-\tau:\,t-1)}), \sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\,t-1)})) = \frac{1}{\sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\,t-1)})\sqrt{2\pi}}\,e^{-\frac{1}{2}\left(\frac{\mathbf{X}_{(t:t+v-1)} - \mu_\phi(\overline{\mathbf{X}}_{(t-\tau:\,t-1)})}{\sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\,t-1)})}\right)^2}$$

We apply logarithm transformation on both sides of the equation,

$$\log \mathcal{N}(\widehat{\mathbf{X}}_{(t:t+v-1)}; \mu_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)}), \sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})) = \log \left[\frac{1}{\sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})\sqrt{2\pi}} \right] + \log \left[e^{-\frac{1}{2}\left(\frac{\mathbf{X}_{(t:t+v-1)} - \mu_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})}{\sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})} \right)^2} \right]$$

$$= \log \frac{1}{\sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})} + \log \frac{1}{\sqrt{2\pi}} - \frac{1}{2}\left(\frac{\mathbf{X}_{(t:t+v-1)} - \mu_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})}{\sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})} \right)^2$$

$$= -\log \sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)}) + C - \frac{1}{2}\left(\frac{\mathbf{X}_{(t:t+v-1)} - \mu_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})}{\sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})} \right)^2$$

We drop the constant(C) and the Gaussian negative log likelihood loss(i.e., negative log gaussian probability density function(pdf)) for the dataset is described by,

$$\mathcal{L}_{\text{GaussianNLLLoss}} = \sum_{t=1}^{\text{T}} \left[\frac{\log \sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})^2}{2} + \frac{(\mathbf{X}_{(t:t+v-1)} - \mu_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)}))^2}{2\sigma_\phi(\overline{\mathbf{X}}_{(t-\tau:\, t-1)})^2} \right]$$

The negative Gaussian log-likelihood is used to evaluate the model's fit for estimated mean and variance to the observations in the training set, where T denotes the time steps. To recapitulate, the objective of the **JHgRF-Net** framework is to minimize the Mean Absolute Error(MAE), while the **w/Unc-JHgRF-Net** framework aims to minimize the $\mathbf{L_{GaussianNLLLoss}}$ to provide quantitative uncertainty estimation. The **w/Unc-JHgRF-Net** framework uses a Gaussian likelihood function to model the mean and variance, optimize the model parameters for data fitting, and provide uncertainty estimates.

8.7. Datasets

Our novel **JHgRF-Net** framework effectiveness was evaluated by comparing it with existing benchmark models on several real-world datasets, including PeMSD3, PeMSD4, PeMSD7, PeMSD7(M), PeMSD8, Electricity, Solar-Energy, Exchange-Rate, Traffic, METR-LA, PEMS-BAY, SWaT, and WADI. Tables 17 - 19 provides additional information regarding these benchmark datasets. The datasets used in this study comprise Solar-Energy, which records solar power production from 137 PV plants in Alabama state at 10-minute intervals in 2016; Electricity, which includes hourly records of electricity consumption(kWh) for 321 clients from 2012 to 2014; Exchange-Rate, which collects daily exchange rates of eight foreign countries from 1990 to 2016; and Traffic dataset provides hourly data on road occupancy rates(ranging from 0 to 1) that were recorded on the various lanes of San Francisco Bay area freeways from 2015 to 2016, spanning 48 months in total. Moreover, METR-LA contains hourly data of traffic speed from loop detectors on the highways in Los Angeles, while PEMS-BAY includes data on traffic volume and speed from sensors on San Francisco Bay area freeways. Additionally, PeMS is an open-access dataset that consists of five traffic network datasets(PeMSD3, PeMSD4, PeMSD7, PeMSD7(M), and PeMSD8), obtained from the Caltrans Performance Measurement System across five California districts, with data points available at 5-minute intervals, providing 288 data points per day. The SWaT dataset consists of 11 days of continuous operation from an industrial water treatment plant, comprising 7 days of normal operation and 4 days with 41 attack scenarios. Two versions of the dataset are available, with Version 1 excluding the first 30 minutes and containing only

normal operation data. The WADI dataset is a collection of 16 days of continuous operation from a testbed, encompassing 14 days of normal operation and 2 days with 15 attack scenarios. An updated version is available that removes affected readings due to plant instability during certain periods.

Table 17
Summary of the traffic-related benchmark datasets.

Dataset	Variables	Timepoints	Time-Range	Split-Ratio	Granularity
PeMSD3	358	26,208	09/2018 - 11/2018		
PeMSD4	307	16,992	01/2018 - 02/2018		
PeMSD7	883	28,224	05/2017 - 08/2017	6 / 2 / 2	5 mins
PeMSD8	170	17,856	07/2016 - 08/2016		
PeMSD7(M)	228	12,672	05/2012 - 06/2012		
METR-LA	207	34,272	03/2012 - 06/2012	7 / 1 / 2	
PEMS-BAY	325	52,116	01/2017 - 05/2017		

Table 18
Summary of the traffic, solar, electricity, and exchange-rate datasets

Dataset	Variables	Timepoints	Granularity	Split-Ratio
Traffic	862	17544	1 hour	
Solar	137	52560	10 min	6/2/2
Electricity	321	26304	1 hour	
Exchange-rate	8	7588	1 day	

Table 19
Summary of the SWaT and WADI datasets

Dataset	Variables	Training Points	Testing Points	Anomalies
SWaT	51	47515	44986	11.97
WADI	127	118795	17275	5.99

8.8. Experimental Study design

In order to examine the effectiveness of the proposed models(**JHgRF-Net**, **w/Unc-JHgRF-Net**) compared to the baselines, the various benchmark datasets were split into training, validation, and test sets. The PEMS-BAY and METR-LA datasets were split with a 7/1/2 ratio, while all other datasets were split with a 6/2/2 ratio. SWaT and WADI datasets have predefined splits, where the training set is anomaly-free and the test set contains anomalies. We utilize the training sets for our experiments. To prepare the SWaT and WADI datasets for training and evaluation, we normalize each variable data by rescaling it to fit within the range of [0, 1], as in [22]. For all other benchmark datasets, each variable data was preprocessed by scaling to have zero mean and unit variance. During the training and evaluation of forecasting models, various accuracy metrics, including MAE, RMSE, and MAPE, were calculated based on the original scale of the time series data. The **JHgRF-Net** architecture was trained for 30 epochs on the training set to minimize forecast error. The validation set was employed to identify the optimal model that improves overall performance and early stopping was utilized to prevent overfitting. The framework performance was evaluated on the test set to examine its ability to perform well on unseen data. To improve convergence, the model training was optimized by using a learning rate scheduler to effectively learn from the training set. If there was no improvement in the evaluation

metrics on the validation set over five epochs, the learning rate was reduced by half. The Adam optimizer was employed to fine-tune the trainable parameters of the models. An initial learning rate of 1×10^{-3} was set to minimize the MAE loss for the **JHgRF-Net** model and the negative Gaussian log-likelihood for the **w/Unc-JHgRF-Net** model, ensuring a better fit between the ground truth and the model predictions. The use of powerful GPUs such as NVIDIA Tesla T4, Nvidia Tesla V100, and GeForce RTX 2080 GPUs expedited the training process and allowed for the utilization of larger models and datasets based on the PyTorch framework. Multiple independent experimental runs were conducted, and the ensemble average was reported to ensure reliable model evaluation. In the Section 8.4, we reported the optimum hyperparameters values of the learning algorithm for each dataset, such as embedding size, number of hyperedges, batch size, and learning rate.

8.9. Baselines

Established algorithms are commonly used as benchmarks for evaluating the performance of proposed neural forecasting models such as **JHgRF-Net** and **w/Unc-JHgRF-Net** on the MTSF task. The selection of benchmark algorithms depends on their extensive usage in the literature and their demonstrated performance on benchmark datasets.

- HA [23] is a time series prediction technique that involves using the average of a predefined historical window of observations to predict the next value in the time series.
- ARIMA is a statistical analysis model commonly used for handling non-stationary time series data, but it has limitations in handling long-term trends or changing seasonal patterns over time.
- VAR([23]) is a linear multivariate time series model that extends the univariate autoregressive(AR) model. It is designed to capture the inter-dependencies among multiple time series variables for analyzing and forecasting complex systems.
- TCN([24]) is specifically designed to handle sequential data in multistep-ahead time series prediction tasks by using causal convolutions and dilation layers to incorporate past information and learn long-range correlations. The use of these techniques allows the model to capture and learn relationships between multiple variables in time series data, making it effective in handling such complex data.
- FC-LSTM([25]) is an encoder-decoder architecture that employs Long Short-Term Memory(LSTM) units with peephole connections to perform multistep-ahead time series prediction. By capturing both short-term and long-term relationships among multiple time series variables in MTS data, this architecture effectively models the intricate patterns and relationships, resulting in a highly complex and nonlinear representation of the data and improved forecasting accuracy.
- GRU-ED([26]) is an encoder-decoder architecture that utilizes Gated Recurrent Unit (GRU) units to handle sequential data in multi-horizon time series prediction tasks. By capturing relevant information from previous time steps, this framework effectively models the sequential dependencies in the data.
- DSANet([27]) is a time series forecasting method that utilizes convolutional neural networks (CNNs) to capture long-range intra-temporal dependencies among multiple

time series, without relying on recurrent networks. In addition, it further incorporates self-attention blocks to adaptively capture interdependencies and generate multi-horizon forecasts for MTS data, resulting in an extremely effective and precise forecasting method.

- DCRNN([28]) is a highly effective technique that combines graph convolution with recurrent neural networks, utilizing bidirectional random walks on graphs. This unique approach enables the model to predict multistep-ahead forecasts in MTS data through an encoder-decoder architecture, which effectively captures complex spatial-temporal dependencies in the data, resulting in highly accurate predictions.

- STGCN([29])is a cutting-edge technique that seamlessly integrates graph convolution and gated temporal convolution networks. By accurately capturing the spatial-temporal correlations among multiple time series variables, this approach enables multi-horizon time series prediction with high precision and accuracy.

- GraphWaveNet([30]) uses a wave-based propagation mechanism and graph representations that are computed from dilated causal convolution neural networks to model MTS data. By jointly learning an adaptive dependency matrix and capturing spatial-temporal dependencies, this method effectively captures the dependencies between multiple time series variables, leading to improved multistep-ahead time series prediction accuracy.

- ASTGCN([31]) utilizes an attention-based spatio-temporal graph convolutional neural network to capture inter- and intra-dependencies for predicting multihorizon forecasts in time series data. By using attention mechanisms, this technique effectively models the spatial-temporal relationships between multiple time series variables, resulting in highly accurate predictions.

- STG2Seq([32]) is an advanced technique for predicting multistep-ahead forecasts in MTS data that incorporates gated graph convolutional networks(GGCNs) with a sequence-to-sequence(seq2seq) architecture featuring attention mechanisms. Through this unique combination, the model captures dynamic temporal and cross-channel information to effectively model the complex relationships among multiple time series variables, resulting in highly accurate and precise predictions.

- STSGCN([33]) is a time series prediction technique that utilizes multiple layers of spatial-temporal graph convolutional networks to predict multistep-ahead forecasts in MTS data. This approach captures localized intra- and inter-dependencies in the graph-structured MTS data, effectively modeling the complex relationships between multiple time series variables and resulting in high precision and accuracy for multi-horizon time series prediction.

- LSGCN([34]) is a method used for multi-horizon time series forecasting, utilizing a graph attention mechanism integrated into a spatial gated block to predict multistep-ahead forecasts in MTS data. By utilizing attention mechanisms, it can effectively model dynamic spatial-temporal dependencies between multiple time series variables, leading to improved forecasting accuracy.

- AGCRN([35]) predicts multistep-ahead forecasts in MTS data by utilizing a data-adaptive graph structure learning method. This approach captures node-specific intra- and inter-correlations, effectively modeling complex spatial-temporal dependencies among the multiple time series variables, resulting in improved forecasting accuracy.

- STFGNN([36]) is a time series prediction technique that fuses representations obtained from temporal graph and gated convolutional neural networks to predict multistep-ahead

forecasts in MTS data. By operating these networks in parallel and learning spatial-temporal correlations, STFGNN effectively models the complex relationships between multiple time series variables, leading to improved forecasting accuracy.

- Z-GCNETs([37]) predicts multi-horizon forecasts in MTS data by integrating a time-aware zigzag topological layer into time-conditioned graph convolutional networks. It captures hidden spatial-temporal dependencies and salient time-conditioned topological information to effectively model complex relationships between multiple time series variables while considering their topological properties. This approach performs well in time series prediction tasks that require the modeling of complex dependencies and relationships.

- STGODE([38]) predicts multistep-ahead forecasts in MTS data using a tensor-based ordinary differential equation (ODE) to capture inter- and intra-dependency dynamics among multiple time series variables. By effectively representing the MTS data, the model can capture the complex relationships among variables and their temporal dynamics, resulting in highly accurate and reliable predictions.

- GDN([22]) is a graph-based anomaly detection model that leverages graph embeddings to learn the inherent complex graph structure underlying the MTS data and further employs a graph forecasting network to compute deviation scores for anomaly detection based on a threshold on forecast error.

- NRI([39]) is an unsupervised technique that learns to deduce interactions and dynamics from observational data within a variational auto-encoder framework. This model effectively predicts the inherent dynamics of complex systems, making it a valuable tool for a wide range of applications.

- MTGNN([3]) presents a graph neural network framework for modeling multivariate time series data. This innovative approach automatically extracts uni-directed relations among multiple time series variables through a graph learning module, while also incorporating mix-hop propagation and dilated inception layers to capture spatial and temporal dependencies within the time series. The result is a highly accurate model capable of multi-horizon forecasts, making it a suitable tool for various time-series applications.

- The vanilla LSTM([?]) predicts multistep-ahead forecasts in MTS data using gating mechanism. The LSTM-U, consisting of N univariate LSTMs, treats all time series variables as independent and performs univariate multi-horizon forecasting.