Non-Aligned Multi-Scale Data Completion for Sparse Mobile CrowdSensing

Wenbin Liu[1,2], Hao Du[1,2], En Wang[1,2], Dongming Luan[1,2], Bo Yang[1,2], Yongjian Yang[1,2], Jie Wu[3]
[1] College of Computer Science and Technology, Jilin University, China
[2] Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, China
[3] China Telecom Cloud Computing Research Institute, China
Email: {liuwb16,duhao22}@mails.jlu.edu.cn, wangen@jlu.edu.cn,
luandm20@mails.jlu.edu.cn, {ybo,yyj}@jlu.edu.cn, wujie@chinatelecom.cn

***Abstract*—Sparse Mobile CrowdSensing has emerged as a practical method for data collection, recruiting mobile users to collect partial data and leveraging spatiotemporal correlations to infer the missing data. To improved the QoS of crowdsourced data, existing methods typically assume that the scales of collected data are similar. However, in real-world scenarios, the diversity of user devices results in data collections that vary in scale. More importantly, the collected coarser-scale data points often do not perfectly correspond to multiple finer-scale data points, resulting in highly complex compositional relationships and posing significant challenges for multi-scale data completion. To address these challenges, this paper proposes a multi-scale data completion framework designed to process and integrate multi-scale data with non-aligned compositional relationships. We first align features across scales using the least common multiple scaling, then enhance the interaction and integration of data across scales through a bidirectional processing strategy and modified Mamba architectures, specifically ST-Mamba and Cross-Mamba. Evaluated on six real-world datasets, our study demonstrates the effectiveness of the proposed framework in handling multi-scale data completion challenges, particularly when dealing with non-aligned compositional relationships.**

***Index Terms*—Mobile CrowdSensing, Quality of Service, multi-scale completion, spatiotemporal data, non-aligned composition.**

I. INTRODUCTION

Sparse Mobile CrowdSensing (Sparse MCS) [1] has emerged as a promising and cost-effective method for data collection in diverse real-world scenarios. Sparse MCS recruits participants equipped with mobile devices, such as smartphones, vehicles, or drones, to collect partial data. Then, data completion, the core of Sparse MCS, utilizes spatiotemporal correlations to reconstruct complete data. This ensures data completeness while reducing collection costs, thereby enhancing Quality of Service (QoS). Currently, Sparse MCS has shown its effectiveness in traffic management [2], environmental monitoring [3], and disaster relief [4].

However, the differences in data collection scales have been significantly neglected by previous research. Existing data completion methods in Sparse MCS, such as compressive sensing [5], matrix completion [6], and deep learning-based techniques [7, 8], often assume that the scales of collected data are the same or similar. Nevertheless, in real-world scenarios, the diversity of user devices often results in data collections that vary in scale [9]. Here, "scale" refers to the sensing range of the devices. As shown in Fig. 1 (left part), consider a scenario where we aim to collect traffic flow data by recruiting three users, each utilizing different devices: a smartphone, a vehicle, and a drone. Obviously, the data collected by the smartphone covers a smaller area, whereas the drone gathers traffic flow data over a much larger range. Such multi-scale data demonstrates significant correlations, such as summation relationships in traffic flow or averaging patterns in air quality, which can enhance data completion. However, existing single-scale methods are inadequate for effectively handling multi-scale sensing data and leveraging the relationships between different scales. Therefore, there is a pressing need for multi-scale data completion methods.

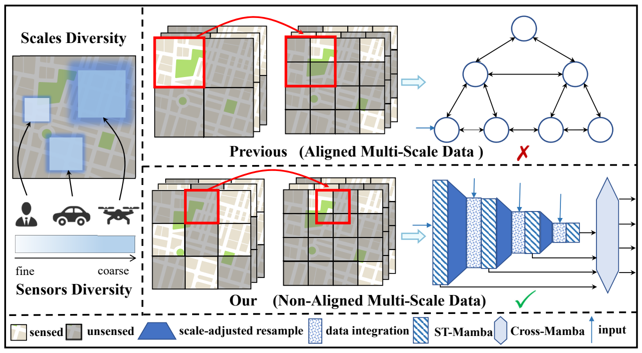


Fig. 1: The Non-Aligned Multi-Scale Scenarios.

To develop an effective multi-scale data completion method for Sparse MCS, three challenges in enhancing QoS arise:

1) How to handle multi-scale data with non-aligned compositional relationships? Recently, a few data mining works, such as Pyraformer [10] and Timemixer [11], have explored the construction and handling of multi-scale data. These studies typically start with single-scale input and artificially create multi-scale data through up- and down-sampling. As shown in Fig. 1 (right part), due to the sampling process ensuring consistent relationships across scales, the created coarse-scale data point is perfectly decomposed into fine-scale data points. This relationship is referred to as *aligned multi-scale data*, which can be structured and processed using a tree-like framework. However, in practical sensing scenarios, the collected coarse-scale data points often do not decompose perfectly into multiple fine-scale data points, referred to as *non-aligned multi-scale data*. This leads to irregular and intricate relationships between data at different scales, posing significant challenges for processing multi-scale data.

2) How to effectively capture intra-scale and inter-scale correlations? Multi-scale data exhibit correlations not only within the same scale (intra-scale) but also across different scales (inter-scale). Intra-scale correlations describe relationships within the same scale, providing insights into localized relationships. Inter-scale correlations, on the other hand, leverage numerical associations across scales, offering additional information for data completion. However, sparse and irregular data lead to incomplete relationships within the same scale, affecting intra-scale correlations by breaking localized patterns. Moreover, non-aligned compositional relationships disrupt the correspondence between scales, making it difficult to directly capture inter-scale correlations. Together, these issues make interactions within and across scales particularly challenging.

3) How to construct a lightweight and effective model for handling multi-scale data? Processing multi-scale data inherently involves higher complexity than single-scale inputs and leads to increased computational overhead. Traditional machine learning methods, despite their low computational costs, primarily address linear spatiotemporal relationships and perform poorly in adapting to multi-scale data processing. Conversely, deep learning methods like Transformers [12] are highly effective and adaptable but lead to significant computational costs. In this context, Mamba [13], an effective and lightweight parallel processing architecture, has attracted our attention. However, its insensitivity to spatiotemporal information limits its applicability in handling multi-scale data.

To address these challenges, we propose a novel framework for non-aligned multi-scale data completion. **1) To handle non-aligned compositional relationships**, we design a Scale-Adjusted Resampling Method based on least common multiple scaling to align multi-scale structures. Additionally, we develop a Spatial Feature Interaction Module (SFIM) to address feature differences across scales after positional alignment. **2) To effectively capture intra-scale and inter-scale correlations**, we separately capture intra-scale and inter-scale correlations, and introduce a bottleneck mechanism in the inter-scale stage to address non-aligned compositional relationships. **3) To construct a lightweight and effective model**, we implement the above interactions using a spatiotemporally sensitive variant of Mamba, called ST-Mamba, for intra-scale processing, and a modified Cross-Mamba for inter-scale information exchange. Both modules ensure linear computational complexity, achieving lightweight and effective processing.

Our work has the following contributions in enhancing QoS of crowdsourced data:

- We investigate the problem of multi-scale data completion in Sparse Mobile CrowdSensing. We propose a novel framework that utilizes sparse data collected at different scales with non-aligned compositional relationships, enabling comprehensive data completion.
- We propose an effective multi-scale data representation method for data completion. We design a Scale-Adjusted Resampling Method to align features across multiple scales. Furthermore, we develop the SFIM to address feature differences across scales after positional alignment.
- We propose the ST-Mamba, a spatiotemporally sensitive Mamba variant for intra-scale feature interaction. Furthermore, inspired by bottleneck techniques, we design the Cross-Mamba for inter-scale feature interaction.
- We evaluate the proposed methods using six real-world datasets, demonstrating that our approach effectively addresses the multi-scale data completion problem.

II. Related Work

A. Sparse Mobile CrowdSensing

Sparse MCS [1] has emerged as a practical solution for QoS data collection, which recruits users to collect partial data and leverages spatiotemporal correlations to infer the remaining ones. Recently, with technological advancements, deep learning-based methods have gradually become the *de facto* choices in Sparse MCS. Wang et al. [6] first introduced a deep learning-enabled approach to enhance QoS for Sparse MCS . Liu et al. [7] also introduced a neural network-based fine-grained data completion and prediction framework. Furthermore, Wang et al. [8] developed a spatiotemporal Transformer model for data inference and long-term prediction in Sparse MCS. However, current algorithms typically assume that data scales are identical or similar, which contradicts the diversity of sensing devices in real-world scenarios. Addressing this discrepancy in multi-scale contexts remains a critical challenge.

B. Multi-scale Model

In recent years, many studies have focused on the analysis of multi-scale features in spatiotemporal data. Liu et al. [10] introduced Pyraformer to enhance prediction accuracy by capturing scale correlations with a pyramid attention mechanism. Similarly, Wang et al. [11] introduced TimeMixer, employing a decomposable multi-scale mixing mechanism to integrate information from various scales. Chen et al. [14] constructed the Multi-Scale Adaptive Graph Neural Network, improving feature extraction and integration by learning multi-scale graph topologies. Additionally, Zhang et al. [15] presented CorrFormer, which uses tree structures to reduce computational complexity while maintaining prediction performance. Despite these advancements, the diversity of sensing devices contributes to challenges with non-aligned multi-scale inputs, underscoring the pressing need for further research in this area.

C. Mamba and Its Variants

Mamba [13] has been proposed as an efficient architecture for handling long-range dependencies in sequential data. Recently, many studies have focused on the application and extension of Mamba. For example, Liu et al. [16] introduced

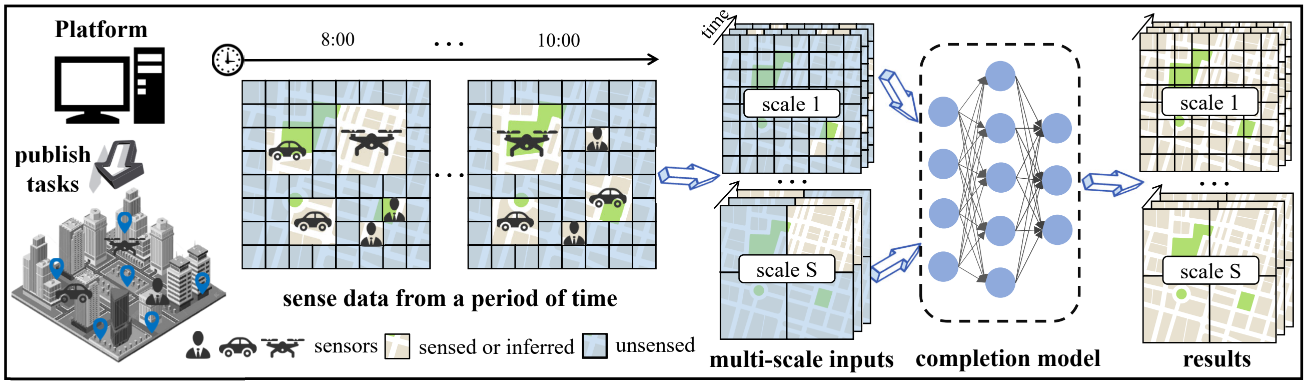


Fig. 2: The workflow of our work: the platform recruits users to collect partial data at multiple scales for data completion.

Vmamba, which employs a visual state space model for efficient visual representation learning. This method incorporates a cross-scan module to traverse the spatial domain, transforming non-causal visual images into ordered block sequences for processing. Zhu et al. [17] developed Vision Mamba, which enhances visual representation learning efficiency by randomly shuffling visual images and utilizing a bidirectional state space model. Additionally, Li et al. [18] proposed STG-Mamba, which leverages a selective state space model for spatiotemporal graph learning. Despite advancements, since Mamba is designed for sequential tasks, its handling of spatial information is limited. Current adaptations struggle with non-intuitive sequential traversals. Enhancing Mamba's spatial sensitivity remains a critical challenge.

III. System model and Problem formulation

A. System Model

In this paper, we consider a comprehensive application scenario of Sparse MCS. The platform recruits users with various devices to collect partial data at specific locations and times, and then integrates the collected data. Ultimately, the platform utilizes these sparse, multi-scale data to achieve complete data collection. To better represent this process, we divide the entire sensed map into a grid with a height of H and a width of W subareas, and the data collection process is segmented into T equal-length time slices.

Scale. We categorize the sensing capabilities of sensors held by users, defined as scales, into S different levels. Each scale, denoted as $g^{(s)}$ for level s, represents the magnification factor relative to the finest scale[1]. For example, a scale of 5 means that each sensing unit covers an area of 5×5 units at the finest scale [2]. This representation helps us to uniformly process and analyze data across different scales. For different scales, we can divide the map into subareas of varying sizes as follows:

$$H^{(s)} = H/g^{(s)}, W^{(s)} = W/g^{(s)}, \quad s = 1, 2, \cdots, S, \tag{1}$$

Data. For the time slice t, the data sensed in the row h and column w of the sensing map at scale level s is defined as $x^{(s)}_{t,h,w}$. Unsensed data is recorded as 0[3]. The true value of this data is denoted as $y^{(s)}_{t,h,w}$, all true values at scale s are denoted as $\mathbf{Y}^{(s)}$. To describe the sensing situation at each scale s, we introduce a sampling matrix $\mathbf{M}^{(s)} \in \mathbb{R}^{T\times H^{(s)}\times W^{(s)}}$, where $m^{(s)}_{t,h,w} = 1$ indicates that data in the $h-$row and $w-$column during the $t-$time slice has been sensed; if $m^{(s)}_{t,h,w} = 0$, it indicates unsensed data. The sensed dataset can be expressed as:

$$\mathbf{X} = \{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \cdots, \mathbf{X}^{(S)}\}, \tag{2}$$

$$\mathbf{X}^{(s)} = \mathbf{Y}^{(s)} \odot \mathbf{M}^{(s)}, \quad s = 1, 2, \cdots, S, \tag{3}$$

where $\mathbf{X}$ represents all the sensed data across different scales, and the dot product ($\odot$) represents the element-wise product.

Method. Utilizing the completion algorithm $\mathcal{I}()$, we input the sensed multi-scale data $\mathbf{X}$ to complete the data at all scales, which is represented as $\hat{\mathbf{Y}}$. Given that data from all other scales can be derived from the finest scale, and because acquiring finer-scale data is inherently more challenging, our primary metric emphasizes minimizing discrepancies at the finest scale. Therefore, we compare $\hat{\mathbf{Y}}^{(1)}$ with the ground truth $\mathbf{Y}^{(1)}$. We denote δ as the error between the completion result and the ground truth at the finest scale, expressed as:

$$\mathcal{I}(\mathbf{X}) = \hat{\mathbf{Y}} \approx \mathbf{Y}, \tag{4}$$

$$\delta(\hat{\mathbf{Y}}^{(1)}, \mathbf{Y}^{(1)}) = \sum\nolimits_{i=1}^{T}\sum\nolimits_{j=1}^{H}\sum\nolimits_{k=1}^{W}\left|y^{(1)}_{i,j,k} - \hat{y}^{(1)}_{i,j,k}\right|. \tag{5}$$

B. Problem Formulation

Problem [Non-Aligned Multi-Scale Data Completion]: Given T time slices, S different scales of data, and $H\times W$ subareas of the finest scale size, we aim to sense data from a limited number of subareas across different scales and utilize this data to reconstruct complete data. We strive to minimize the error between the completion results and the ground truth.

$$\min\ \delta(\hat{\mathbf{Y}}^{(1)}, \mathbf{Y}^{(1)}) = \sum\nolimits_{i=1}^{T}\sum\nolimits_{j=1}^{H}\sum\nolimits_{k=1}^{W}\left|y^{(1)}_{i,j,k} - \hat{y}^{(1)}_{i,j,k}\right|. \tag{6}$$

[1]To simplify matters, we assume that all scales inherently contain the finest scale. Our method can be easily adapted to other scenarios by assuming the finest scale is the greatest common denominator of all scales.

[2]To simplify matters, we assume each sensing unit covers a square area.

[3]If 0 has a specific meaning, an alternative value will be used.

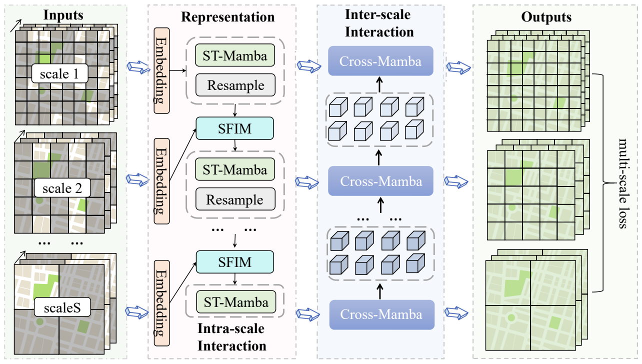


Fig. 3: The structure of our model.

C. Workflow

Our workflow is shown in Fig. 2. Consider a real-word example, platform publishes an urban data sensing task and recruits five users for data collection. Two of these users employ smartphones (fine-scale), two use vehicles (mid-scale), and one operates a drone (coarse-scale). The platform assigns specific times and locations for data collection based on the scale of each user's device. Following these directives, the five users collect and upload their data. The platform aggregates the data across different scales and feeds the spatiotemporal maps corresponding to each scale into the completion model, ultimately generating completion results for all scales.

IV. Method

A. Overall Structure

As illustrated in Fig. 3, our model consists of four components: Data Representation, Intra-scale Feature Interaction, Inter-scale Feature Interaction, and Output Section. Initially, we conduct preliminary data embedding across all scales. Subsequently, scale-aligned resampling is utilized to construct new scale features from finer-scale features. These newly constructed features are then merged with sparse input features for each scale via the SFIM, to address feature differences. Following this, ST-Mamba is employed to process these integrated features for intra-scale interaction. After processing features from all scales, they are input into the Inter-scale Feature Interaction component, where the Cross-Mamba mechanism analyzes the relationships among different scales. Finally, we use the output layer to get results, and the network is trained based on the multi-scale loss.

B. Multi-scale Data Representation

1) Data Embedding: Data embedding is crucial for model's performance and accuracy, especially when handling multi-scale spatiotemporal data. Our embedding module integrates value embedding, position embedding, spatial information (latitude, longitude, POI) embedding, and temporal information (timestamp, holiday) embedding. Furthermore, to enhance the model's responsiveness to different scales, we design different parameters for inputs of different scales, while inputs at the same scale share the same parameters. This approach improves the model's adaptability to multi-scale inputs, ensuring that it

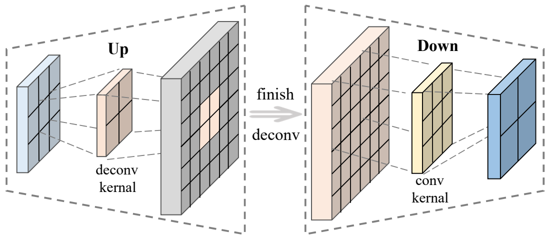


Fig. 4: The example of scale-adjusted convolution.

can effectively handle the diverse and dynamic characteristics of spatiotemporal data across various scales.

The embedding of scale s $\mathbf{X}_{emb}^{(s)} \in \mathbb{R}^{T\times H^{(s)}\times W^{(s)}\times d_{model}}$ can be expressed as:

$$\mathbf{X}_{emb}^{(s)} = \mathbf{X}_{val}^{(s)} + \mathbf{X}_{p}^{(s)} + \mathbf{S}_{f}^{(s)} + \mathbf{T}_{f}^{(s)}, \tag{7}$$

where $\mathbf{X}_{val}^{(s)}$ represents value embedding, $\mathbf{X}_{p}^{(s)}$ represents spatiotemporal position embedding, $\mathbf{S}_{f}^{(s)}$ represents spatial embedding, and $\mathbf{T}_{f}^{(s)}$ represents temporal embedding.

2) Scale-Adjusted Resampling Method: Constructing multi-scale features is crucial in multi-scale research, and an effective method can significantly boost model efficiency. Our goal is to capture the multi-scale characteristics of spatiotemporal data across both spatial and temporal dimensions.

In the spatial dimension, the diversity of sensing devices introduces variability in spatial scales. Standard convolutional methods often struggle with non-aligned compositional relationships between scales. For example, downsampling a 5×5 data matrix to 3×3 typically involves a convolution operation with a stride of 2, kernel size of 2, and padding of 1. However, ideally, each 3×3 output cell should correspond to an area of approximately $5/3\times 5/3$ in the original matrix. Although the traditional convolution operation can ensure the correct output size, it cannot achieve this non-integer mapping, leading to misalignments. These misalignments can cause significant information loss and challenges in feature alignment, adversely impacting subsequent analyses.

To address this issue, we develop a novel scale-adjusted convolution approach designed to ensure effective alignment between different scales. This method includes an initial upsampling step to spatially align coarser-scale data with finer-scale data, followed by a downsampling step to match the dimensions of the target scale. As illustrated in Fig. 4, to downsample from 3×3 to 2×2, we first upsample the 3×3 data to 6×6 (their least common multiple size). Then, we apply a standard convolution to downsample from 6×6 to the target size of 2×2. This process can be expressed as:

$$\mathbf{X}_{conv}^{(s)} = \mathrm{relu}(\mathrm{conv}(\mathrm{relu}(\mathrm{deconv}(\mathbf{X}_{rep}^{(s-1)})))), \tag{8}$$

where $\mathbf{X}_{rep}^{(s-1)}$ represents the features from scale $s-1$. In addition, to enrich features, we utilize both pooling and convolution results in constructing the coarse-scale features. Here, we employ AdaptivePool to ensure the pooled output is aligned to the required dimensions, expressed as:

$$\begin{aligned}\mathbf{X}_{pool}^{(s)} &= \mathrm{AdaptivePool}(\mathbf{X}_{rep}^{(s-1)}),\\ \mathbf{X}_{down}^{(s)} &= \mathrm{Linear}(\mathrm{concat}(\mathbf{X}_{pool}^{(s)}, \mathbf{X}_{conv}^{(s)})).\end{aligned} \tag{9}$$

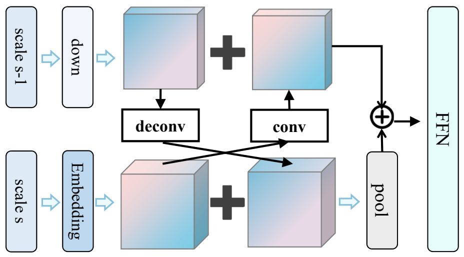


Fig. 5: The structure of Sparse Feature Integration Module.

In the temporal dimension, as our data collection follows fixed time intervals, there is no issue of non-alignment. In this case, we adopt a method similar to traditional time series analysis by downsampling with a stride of 2 for temporal. These techniques allow us to effectively process spatial scales and capture connections between long-term and short-term data, thus enhancing the depth of our multi-scale data analysis.

3) Sparse Feature Integration Module (SFIM): After re-sampling, the newly constructed features and the originally collected sparse data often exhibit significant differences. These differences, arising from variations in data scale and information content, hinder the effective utilization of both types of features. In this respect, we propose SFIM to fuse these two types of features, enabling comprehensive data utilization.

Since the two inputs contain information of two different distributions, simple fusion may lead to poor learning and ultimately impair model performance. To fully explore and leverage the association between these two features, we design a bidirectional processing strategy similar to the one in [19]. As illustrated in Fig. 5, because the features constructed represent a coarser time scale, and the input features represent a finer time scale, we use deconvolution to upsample the features constructed from the previous scale, enhancing their time resolution to better match the input fine-scale features. At the same time, we use convolution to downsample the direct input fine-scale features to align with the constructed coarse-scale features. Then, we fuse these two types of features:

$$\mathbf{T}_{down}^{(s)} = \mathrm{deconv}(\mathbf{X}_{down}^{(s)}), \mathbf{Z}_{down}^{(s)} = \mathbf{T}_{down}^{(s)} + \mathbf{X}_{emb}^{(s)}, \quad (10)$$

$$\mathbf{T}_{emb}^{(s)} = \mathrm{conv}(\mathbf{X}_{emb}^{(s)}), \mathbf{Z}_{emb}^{(s)} = \mathbf{T}_{emb}^{(s)} + \mathbf{X}_{down}^{(s)}. \quad (11)$$

Additionally, to further optimize the feature integration process, we introduce a weight matrix $\mathbf{W}$, calculated through a Feed Forward Neural Network (FFN) that combines the previously fused two types of features, outputting a weight value to dynamically adjust the contribution of each scale feature in the final integration output.

$$\mathbf{W} = \sigma(\mathrm{FFN}(\mathrm{pool}(\mathbf{Z}_{down}^{(s)}) + \mathbf{Z}_{emb}^{(s)})), \quad (12)$$

$$\mathbf{X}_{rep}^{(s)} = \mathbf{W} \odot \mathbf{Z}_{down}^{(s)} + (1 - \mathbf{W}) \odot \mathbf{Z}_{emb}^{(s)}. \quad (13)$$

where ($\odot$) denotes element-wise multiplication and σ denotes the sigmoid function. This method enables us to adaptively adjust the impact of features based on their information and importance, thereby maximizing the utilization of limited information while minimizing information loss.

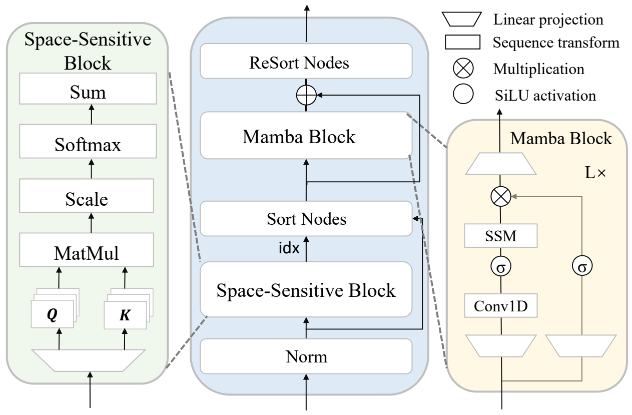


Fig. 6: The structure of ST-Mamba model.

C. Intra-scale Feature Interaction

An important issue in dealing with multi-scale data is to capture intra-scale information correlations. Currently, the attention-based architectures, such as Transformers, are commonly employed to capture these relationships due to their comprehensive capabilities. However, the quadratic complexity of Transformers leads to substantial computational overhead. While some attempts have been made to address this issue by deploying lighter variants of Transformers, these efforts often result in reduced effectiveness. Consequently, Mamba has emerged as a promising alternative due to its potential for efficiency. However, Mamba lacks spatiotemporal sensitivity, which we address with our ST-Mamba design.

1) Mamba: Mamba is a lightweight model designed to handle sequential problems, essentially an improvement on State Space Models (SSM) [20]. SSM is based on a continuous system that models the evolution of internal states over time. The continuous system can be expressed as:

$$h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t), \ y(t) = \mathbf{C}h(t), \quad (14)$$

where $h(t) \in \mathbb{R}^N$ represents hidden state, $\mathbf{A} \in \mathbb{R}^{N\times N}$ represents evolution parameter, $\mathbf{B} \in \mathbb{R}^{N\times 1}$ and $\mathbf{C} \in \mathbb{R}^{N\times 1}$ represent projection parameters. To make the system applicable to real-world scenarios, the continuous system is discretized by the Zero-Order Hold method. This transformation converts the continuous parameters $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ into discrete counterparts:

$$\bar{\mathbf{A}} = \exp(\Delta\mathbf{A}), \bar{\mathbf{B}} = (\Delta\mathbf{A})^{-1}(\exp(\Delta\mathbf{A}) - \mathbf{I}) \cdot \mathbf{B}, \quad (15)$$

$$h_t = \bar{\mathbf{A}}h_{t-1} + \bar{\mathbf{B}}x_t, y_t = \mathbf{C}h_t. \quad (16)$$

where Δ represents the sampling time interval.

SSM models are designed for linear time-invariant scenarios, however, many real-world scenarios are non-linear time-variant. To overcome this limitation, Mamba introduces an input-dependent selection mechanism that enables effective information filtering from the input. Specifically, the parameters $\mathbf{B} \in \mathbb{R}^{B\times L\times N}$, $\mathbf{C} \in \mathbb{R}^{B\times L\times N}$ and $\Delta \in \mathbb{R}^{B\times L\times N}$ are directly derived from the input data $\mathbf{x} \in \mathbb{R}^{B\times L\times N}$, thereby enhancing adaptability and processing efficiency in dynamic environments. Mamba also introduces the selective scan method to

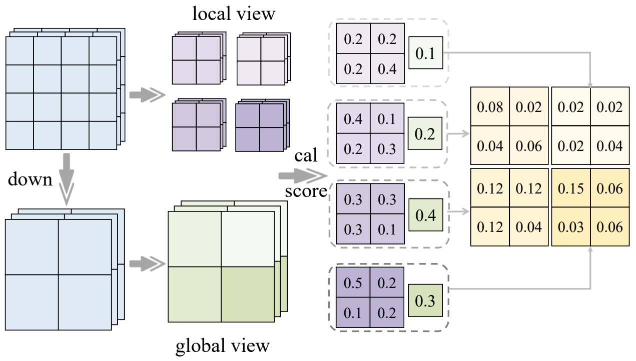


Fig. 7: The structure of importance score calculation method with complexity optimization.

improve data scanning and processing efficiency, significantly boosting inference speed, which can be expressed as:

$$\overline{\mathbf{K}} = (\mathbf{C}\overline{\mathbf{B}}, \mathbf{C}\overline{\mathbf{A}}\overline{\mathbf{B}}, \dots, \mathbf{C}\overline{\mathbf{A}}^{\mathbf{L}-1}\overline{\mathbf{B}}), \mathbf{y} = \mathbf{x} * \overline{\mathbf{K}}, \tag{17}$$

where $\overline{\mathbf{K}}$ is a structured convolutional kernel.

2) ST-Mamba: Our task involves handling spatiotemporal data, making the capture of spatiotemporal correlations crucial. [21] has demonstrated the feasibility of applying Mamba to time series data, effectively capturing temporal dependencies. However, since Mamba is designed for sequential tasks, its architecture is insensitive to spatial information. To address this issue, we have improved the original Mamba architecture to better capture spatial correlations.

In sequential models like Mamba, later nodes are inherently more important as they incorporate the receptive fields of preceding nodes. This property works well for temporal sequences where the order of data collection is linear and sequential. However, in the context of spatiotemporal data, the spatial arrangement of nodes is not naturally ordered, posing a challenge in effectively capturing spatial correlations.

To tackle this issue, we propose the ST-Mamba architecture. As illustrated in Fig. 6, we first input the data into a Space-Sensitive Block to determine the importance of spatial nodes. After sorting the nodes based on their spatial importance, we feed the ordered data into Mamba. Within the Space-Sensitive Block, we utilize an attention mechanism to calculate the scores of nodes based on their spatial relevance. Here, we employ spatial attention, which focuses solely on nodes within the same time slice, ensuring that temporal order is preserved and that the computational complexity remains linear with respect to time[4]. This process can be expressed as:

$$\mathbf{Q}_{Sp}^{(s)} = \mathbf{X}_{rep}^{(s)}\mathbf{W}_{Sp}^{Q}, \ \mathbf{K}_{Sp}^{(s)} = \mathbf{X}_{rep}^{(s)}\mathbf{W}_{Sp}^{K}, \tag{18}$$

$$\mathbf{attn}_{Sp}^{(s)} = \text{softmax}(\mathbf{Q}_{Sp}^{(s)}\mathbf{K}_{Sp}^{(s)}/\sqrt{d_k}), \tag{19}$$

where $\mathbf{W}_{Sp}^{Q}$ and $\mathbf{W}_{Sp}^{K}$ are learnable weights for the spatial dimensions, and $\sqrt{d_k}$ is the dimension of $\mathbf{K}_{Sp}^{(s)}$. Unlike simply stacking attention and Mamba blocks, our method uses attention scores to assign spatial order to the nodes, effectively

[4]In practical applications, as the sensed map is fixed, the number of spatial points remains constant and can be viewed as a fixed value.

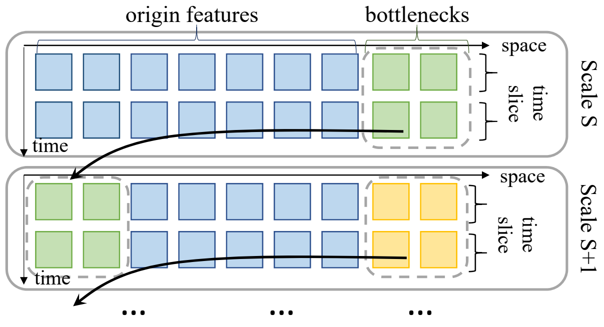


Fig. 8: The example of Cross-Mamba.

enhancing spatial information integration. This method not only retains the lightweight advantage of the Mamba model but also enhances its sensitivity to spatial information, making it more efficient and accurate in handling spatiotemporal data.

3) Complexity Optimization: Due to the large number of spatial nodes at the finest scale, the computational overhead remains significant. For this purpose, we simplify the finest scale calculation process. As illustrated in Fig. 7, we first downsample the data to obtain global features and calculate the global scores. For the finest-scale nodes within the corresponding coarse-scale nodes, we compute their local scores. The final score of a node at the current scale is the product of the global score of its coarse-scale region and its local score. Finally, we use a learnable weight to perform a weighted average of the scores across different scales, as detailed below:

$$\begin{aligned} \mathbf{score}^{(s)} &= \text{Global}^{(s)}(\mathbf{X}_{rep}^{(1)}) \times \text{Local}^{(s)}(\mathbf{X}_{rep}^{(1)}), \\ \mathbf{score} &= \sum\nolimits_{s=2}^{S} \mathbf{score}^{(s)} \times \mathbf{w}_{s-1}, \end{aligned} \tag{20}$$

where $\text{Global}^{(s)}$ and $\text{Local}^{(s)}$ denote the global and local score calculation methods from the perspective of scale s, and $\mathbf{w}$ represents the learnable weight. Ultimately, we denote the intra-scale interaction features at scale s as $\mathbf{X}_{\text{intra}}^{(s)}$:

$$\mathbf{X}_{\text{intra}}^{(s)} = \text{ST-Mamba}(\mathbf{X}_{\text{rep}}^{(s)}). \tag{21}$$

D. Inter-scale Feature Interaction

Capturing correlations between multiple scales in multi-scale tasks is crucial. Unlike the Transformer architecture, which achieves direct interaction through cross-attention, Mamba struggles with direct interactions between multiple data scales. Simply inputting all scale data into a single Mamba model poses two problems: determining the order of multi-scale information and incurring significant computational overhead. Inspired by the bottleneck architecture [22], we integrate this structure into Mamba, called Cross-Mamba.

The bottleneck architecture significantly reduces computational cost while preserving the main features of the data. As illustrated in Fig. 8, since the later nodes in the Mamba architecture can learn from the preceding nodes, we append additional empty nodes (bottlenecks) to the original sequence and input them into ST-Mamba to enable the learning of global receptive fields. Subsequently, we prepend the learned content to finer-scale nodes, allowing them to learn the receptive fields of coarser scales. Similarly, the final empty nodes obtain global receptive fields, and this process is repeated down to the finest scale. We place bottleneck nodes at the end of each time dimension to capture all spatial information within each time slice:

$$[\mathbf{X}^{(1)}_{\text{inter}}, \mathbf{E}^{(1)}] = \text{STM}\left([\mathbf{X}^{(1)}_{\text{intra}}, \mathbf{B}^{(1)}]\right), \tag{22}$$

$$[\_, \mathbf{X}^{(s)}_{\text{inter}}, \mathbf{E}^{(s)}] = \text{STM}\left([\mathbf{E}^{(s-1)}, \mathbf{X}^{(s)}_{\text{intra}}, \mathbf{B}^{(s)}]\right), \tag{23}$$

$$[\_, \mathbf{X}^{(S)}_{\text{inter}}] = \text{STM}\left([\mathbf{E}^{(S-1)}, \mathbf{X}^{(S)}_{\text{intra}}]\right), \tag{24}$$

where _ represents placeholder, STM represents ST-Mamba method, $\mathbf{B}^{(s)}$ represents bottleneck nodes of scale s, $\mathbf{E}^{(s)}$ represents the learned information, and $\mathrm{s} \in [2, \mathrm{S}-1]$. This ensures a comprehensive learning process, leveraging the bottleneck structure to effectively capture multi-scale correlations.

E. Multi-scale Loss

For each scale, we use a linear layer to obtain the final completion results and consider the completion effect of each scale during the final loss calculation. Formally:

$$\hat{\mathbf{Y}}^{(s)} = \text{Linear}(\mathbf{X}^{(s)}_{\text{inter}}), \tag{25}$$

$$\mathcal{L} = \sum_{s=1}^{S} C_s \times \text{MSE}(\hat{\mathbf{Y}}^{(s)}, \mathbf{Y}^{(s)}), \tag{26}$$

where C_s is a constant denoting the weight of the s-th scale result. We show our whole workflow in Algorithm 1.

Algorithm 1: The algorithm of our method.

```
Input: $\mathbf{X}$ - sparse multi-scale inputs
1  while not convergent and count < MAX_ITER do
2      Data Representation:
3      $\mathbf{X}^{(1)}_{emb} \leftarrow$ Eq.(7)($\mathbf{X}^{(1)}$);
4      $\mathbf{X}^{(1)}_{rep} \leftarrow \mathbf{X}^{(1)}_{emb}$;
5      for each $s$ in $[2, S]$ do
6          $\mathbf{X}^{(s)}_{emb} \leftarrow$ Eq.(7)($\mathbf{X}^{(s)}$);
7          $\mathbf{X}^{(s)}_{down} \leftarrow$ Eq.(9)($\mathbf{X}^{(s-1)}_{rep}$);
8          $\mathbf{X}^{(s)}_{rep} \leftarrow$ Eq.(12)($\mathbf{X}^{(s)}_{emb}, \mathbf{X}^{(s)}_{down}$);
9      Intra-scale Feature Interaction:
10     for each $s$ in $[1, S]$ do
11         $\mathbf{X}^{(s)}_{intra} \leftarrow$ Eq.(21)($\mathbf{X}^{(s)}_{rep}$);
12     Inter-scale Feature Interaction:
13     $[\mathbf{X}^{(1)}_{inter}, \mathbf{E}^{(1)}] \leftarrow$ Eq.(22)($[\mathbf{X}^{(s)}_{intra}, \mathbf{0}]$);
14     for each $s$ in $[2, S-1]$ do
15         $[\mathbf{X}^{(s)}_{inter}, \mathbf{E}^{(s)}] \leftarrow$ Eq.(23)($[\mathbf{E}^{(s-1)}, \mathbf{X}^{(s)}_{intra}, \mathbf{0}]$);
16     $\mathbf{X}^{(S)}_{inter} \leftarrow$ Eq.(24)($[\mathbf{E}^{(S-1)}, \mathbf{X}^{(s)}_{intra}]$);
17     Output:
18     for each $s$ in $[1, S]$ do
19         $\hat{\mathbf{Y}}^{(s)} \leftarrow$ Eq.(25)($\mathbf{X}^{(s)}_{inter}$);
20     $\mathcal{L} \leftarrow$ Eq.(26)($\hat{\mathbf{Y}}, \mathbf{Y}$);
21 return $\hat{\mathbf{Y}}^{(1)}$.
```

V. THEORETICAL ANALYSIS

We prove the necessity of multi-scale data utilization in Theorem 1 and the complexity of ST-Mamba in Theorem 2.

Definition 1. *Let $\mathbf{X}_{\text{multi}}$ denote the multi-scale data representation information, and let $\mathbf{X}_{\text{single}}$ denote the input data information at only single scale.*

Theorem 1. *The mutual information between $\mathbf{X}_{\text{multi}}$ and the target variable $\mathbf{Y}$ consistently exceeds the mutual information between $\mathbf{X}_{\text{single}}$ and $\mathbf{Y}$.*

Proof. Mutual information is defined as follows:

$$\begin{aligned} I(\mathbf{Y}; \mathbf{X}_{\text{multi}}) &= H(\mathbf{Y}) - H(\mathbf{Y} \mid \mathbf{X}_{\text{multi}}), \\ I(\mathbf{Y}; \mathbf{X}_{\text{single}}) &= H(\mathbf{Y}) - H(\mathbf{Y} \mid \mathbf{X}_{\text{single}}), \end{aligned} \tag{27}$$

where $H(\mathbf{Y})$ denotes the entropy of $\mathbf{Y}$, and $H(\mathbf{Y} \mid \mathbf{X}_{\text{multi}})$ and $H(\mathbf{Y} \mid \mathbf{X}_{\text{single}})$ represent the conditional entropy given multi-scale and single-scale data, respectively.

Since $\mathbf{X}_{\text{multi}}$ is constructed by integrating data from multiple scales, it inherently encapsulates all the information in $\mathbf{X}_{\text{single}}$ and additional features from other scales. This additional information reduces the uncertainty of the target variable $\mathbf{Y}$, leading to:

$$H(\mathbf{Y} \mid \mathbf{X}_{\text{multi}}) \le H(\mathbf{Y} \mid \mathbf{X}_{\text{single}}). \tag{28}$$

Combining Eq. (27) and Eq. (28), we obtain:

$$I(\mathbf{Y}; \mathbf{X}_{\text{multi}}) \ge I(\mathbf{Y}; \mathbf{X}_{\text{single}}), \tag{29}$$

indicating that the multi-scale representation improves the mutual information with the target variable $\mathbf{Y}$ compared to single-scale data. This result highlights the importance of utilizing multi-scale data for completion. □

Definition 2. *Denote $L^{(s)} = H/g^{(s)} \times W/g^{(s)}$ as the number of spatial nodes, T as the number of time slices.*

Theorem 2. *The computational complexity of ST-Mamba is linear with respect to the number of time slices T.*

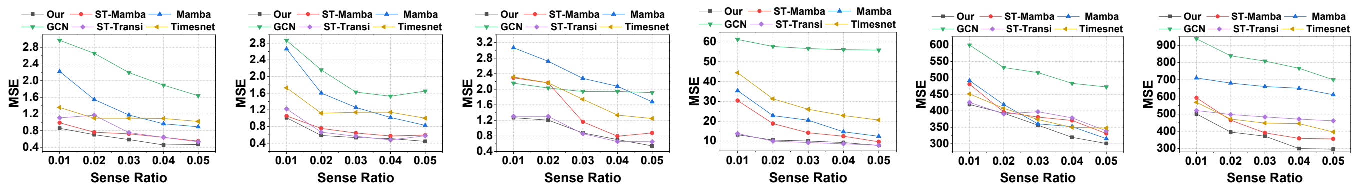
Proof. ST-Mamba is composed of two main parts: the spatial sensitivity module and the Mamba processing unit. In the spatial sensitivity module, for each scale s (except the finest scale), the computational complexity is governed by the interactions within each time slice. This results in a complexity $T \times \left(L^{(s)}\right)^2$. The finest scale is calculated differently, following Eq. (20), its computational complexity is: $\sum_{s=2}^{S}\left[T \times \left(L^{(s)}\right)^2\right]$. Adding the linear Mamba complexity for processing all scales, which is $\sum_{s=1}^{S} T \times L^{(s)}$. The total complexity becomes:

$$T \times \left[2 \times \sum_{s=2}^{S} \left(L^{(s)}\right)^2 + \sum_{s=1}^{S} L^{(s)}\right]. \tag{30}$$

Since the sensed map is deterministic in real scenarios, $L^{(s)}$ can be treated as a constant. Thus the complexity of ST-Mamba is linearly related to T. □

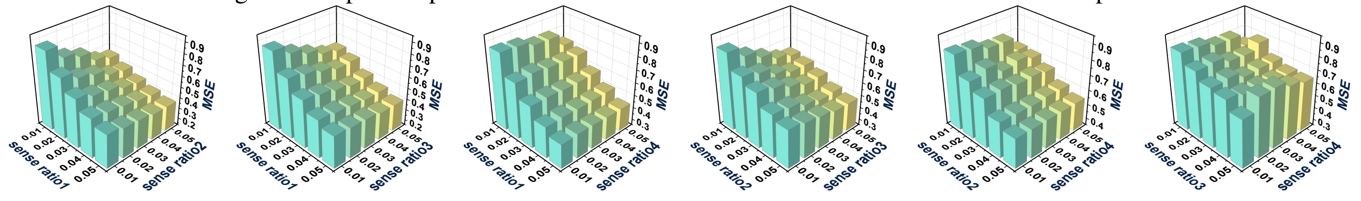
TABLE I: Completion performance under the same sensed quantity.

Data Set	Air-Quality				Weather				Traffic			
	PM2.5		PM10		Humidity		Temperature		P1		P2	
Models	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
MF	60.070	7.000	60.146	7.000	33.601	4.465	363.61	16.543	6250.6	42.251	8623.4	49.746
DMF	69.753	6.175	68.866	6.134	57.170	5.738	351.84	13.621	14660.8	69.171	18115.0	76.284
GCN	2.969	1.545	2.865	1.520	2.158	1.064	61.313	5.466	599.94	14.880	938.05	17.785
Mamba	2.220	0.945	2.662	1.008	3.069	1.220	35.419	4.048	491.85	12.914	710.42	15.594
Informer	2.526	1.288	3.374	1.506	2.949	1.294	47.384	4.984	534.42	13.717	627.17	14.207
Timesnet	1.359	0.855	1.729	1.005	2.317	1.111	44.491	4.837	451.28	13.196	568.56	14.799
TS-BGMC	3.589	1.424	3.585	1.426	22.33	3.565	384.365	15.799	9452.02	51.930	12601.53	60.269
ST-Transi	1.108	0.765	1.221	0.802	1.306	0.873	13.782	2.665	425.81	12.214	520.97	13.710
ST-Mamba	0.989	0.703	1.053	0.740	2.294	1.086	30.410	3.812	480.95	12.804	594.77	14.281
Our	**0.857**	**0.653**	**1.009**	**0.723**	**1.269**	**0.859**	**13.179**	**2.616**	**419.18**	**12.043**	**500.97**	**13.510**



(a) PM2.5 (b) PM10 (c) Temperature (d) Humidity (e) P1 (f) P2

Fig. 9: Completion performance under different sensed ratios with multi-scale inputs.



(a) scale1,scale2 (b) scale1,scale3 (c) scale1,scale4 (d) scale2,scale3 (e) scale2,scale4 (f) scale3,scale4

Fig. 10: Completion performance under two different scales.

VI. Performance Evaluation

A. Setting

1) Datasets: We conduct experiments on three datasets.

Air-Quality[5] is derived from CHAP [23, 24] and contains meteorological air quality data. We select PM2.5 and PM10 as the experimental datasets and select 20×20 stations.

Weather[6] contains meteorological data from 2017 to 2018. We select Humidity and Temperature as the experimental datasets and select 24×12 stations for experiments.

Traffic[7] is derived from TaxiBJ [25], which contains meteorological traffic flow data in Beijing. We select P1 and P2 as the experimental datasets and select 16×16 stations.

2) Data Preprocessing: We set the total number of scales to 4. To demonstrate the adaptability of our proposed method to real-world scenarios with non-aligned compositions, we use different scales for different datasets. Specifically, the scales for the Air-Quality dataset are set to 1, 2, 4, and 5, and for the Weather dataset, they are 1, 2, 3, and 6. Additionally, to prove that our approach is also effective in aligned scenarios, the scales for the Traffic dataset are set to 1, 2, 4, and 8.

We preprocess the data to obtain results at different scales. Taking the scale of 2 as an example, we downsample the original data by a factor of 2×2. The downsampling rules follow the inherent patterns of the data: for Air-Quality and Weather use averaging, while Traffic for summation.

3) Baselines: We categorize the existing methods into three categories: temporal interpolation, spatial completion and spatiotemporal completion methods.

Spatial Completion Methods: We consider MF [26], which is based on matrix factorization; DMF [27], a neural network implementation of matrix factorization; and GCN [28], which utilizes a graph convolutional network approach.

Temporal Interpolation Methods: We consider Mamba [13], a linear-time sequence modeling method; Informer, which employs sparse attention mechanisms [29]; and Timesnet [30], a general approach to time series embedding.

Spatiotemporal Completion Methods: We analyze methods like ST-BGMC [7], a matrix completion technique that utilizes spatiotemporal constraints; ST-Transi [8], a Transformer-based method; and our own proposal, ST-Mamba, a variant of Mamba model designed to handle spatiotemporal data.

4) Experimental Settings: In our experiment, the datasets are divided into training, validation, and test sets in a $7:2:1$ ratio. We use Mean Squared Error (MSE) and Mean Absolute Error (MAE) as our evaluation metrics. All experiments are conducted under PyTorch and accelerated using a single NVIDIA GeForce RTX 3090 GPU. In our model d_{model} is set to 64, T is set to 16, model layer is set to 1 and bottleneck size is set to 4. To better complement the results at finer scales, we give them more weight, so C is set to $\{0.4, 0.3, 0.2, 0.1\}$.

[5]https://nnu.geodata.cn/featured_data.html
[6]https://www.kdd.org/kdd2018/kdd-cup
[7]https://gitee.com/arislee/taxi-bj

B. Completion Performance

1) Performance under the same sensed quantity: We conduct experiments with a sensing rate of 0.01 for each scale input. Since existing completion methods are designed for single-scale, we ensure a fair comparison by inputting the finest scale data with the total sensing amount from all scales into other models. As shown in TABLE I, our method significantly outperforms the others. In real-world multi-scale scenarios, the worse performance of single-scale methods highlighting the effectiveness of leveraging multi-scale data.

2) Performance under the finest scale sense ratios: We evaluate model completion effects under different finest scale sensing ratios, with a 0.01 ratio for other scales. The number of sensed data for other models matches the total number from all scales in our method. As shown in Fig. 9, our multi-scale model excels due to its ability to capture multi-scale data correlations. Although it sometimes lags behind Transformer-based models, it has significantly lower overheads. For Air-Quality and Weather datasets, high spatial similarity enhances spatiotemporal algorithm performance. For the Traffic dataset, stronger temporal correlations result in similar performance between spatiotemporal and temporal algorithms, causing overlap in the graph. Additionally, comparisons with ST-Mamba demonstrate the effectiveness of Cross-Mamba.

3) Performance under multi-scale sense ratios: Fig. 10 further explores the impact on model completion performance as the sense ratios of two scales are increased. Due to experimental constraints, this part of the study is only conducted on the PM2.5 dataset. The results show that the overall performance of the model tends to increase as the scale increases and that finer scales provide better results for the same sensing range.

C. Ablation Study

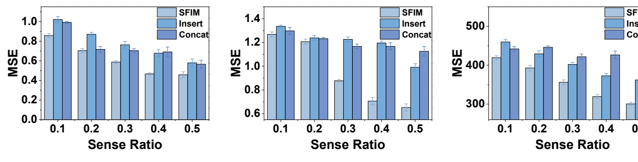
Due to the space constraints, we use PM2.5, Humidity and P1 as the representative datasets for our experiments.

TABLE II: Performance of different resample methods.

Method	Air		Weather		Traffic	
	MSE	MAE	MSE	MAE	MSE	MAE
scale-adjusted+pool	**0.857**	**0.653**	**1.269**	**0.859**	**419.177**	**12.043**
scale-adjusted	0.877	0.673	1.292	0.873	448.466	12.157
pool	0.975	0.717	1.512	0.949	464.561	12.377
conv	0.974	0.716	1.452	0.930	497.299	12.579

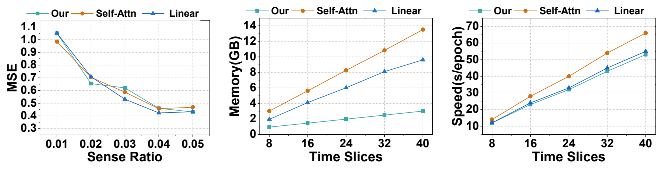
1) Resample method: We compared the performance of mainstream resampling methods with our proposed up-down sampling approach. As shown in TABLE II, our method, which considers the alignment of scale features, significantly outperforms other methods. Additionally, considering both pooling and convolution information yields superior results.

2) Feature integration module: We compared our SFIM with two common methods, Insert and Concat. Specifically, Insert involves directly filling the sparse input features into the downsampled features, while Concat involves concatenating both feature sets followed by a linear layer for feature fusion. The results, as illustrated in Fig. 11, show superior performance with our approach. This improvement is attributed to the distinct data distributions in the two inputs; simplistic approaches tend to allow these distributions to interfere with each other, leading to a decrease in performance.



(a) Air (b) Weather (c) Traffic

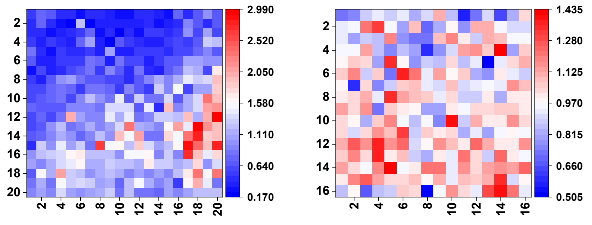
Fig. 11: Performance of feature integration module.



(a) Performance (b) Memory (c) Speed

Fig. 12: Performance and cost of Space-Sensitive Block.

3) Space-Sensitive Block: Through previous experiments, we have demonstrated the superior performance of our proposed ST-Mamba compared to the origin Mamba, highlighting the effectiveness of our space-sensitive module. Therefore, this section will not redundantly prove the effectiveness of this method through further experiments. Instead, we will conduct detailed experiments on the complexity optimization of our space-sensitive module, examining both performance and overhead. We conducted comparative analyses among linear scoring [31] and original self-attention scoring. As shown in Fig. 12, our approach effectively reduces the model's complexity without causing significant performance loss.



(a) Air (b) Traffic

Fig. 13: Visualization of spatial importance.

D. Spatial Sensitivity Analysis

We have shown the importance of spatial nodes for air and traffic. As illustrated in the Fig. 13, the importance scores for air quality data are notably concentrated in specific urban areas, likely indicating regions with significant environmental pressures or monitoring demands, such as industrial activity zones. Conversely, the importance scores for traffic flow data are more uniformly distributed across the entire city, suggesting a widespread impact and connectivity of the urban traffic system. These differences suggest that our model can be responsive to different types of urban data.

TABLE III: Computing overhead for main methods.

	Mamba	**Transformer**	**Our**
running time	22 s/epoch	246 s/epoch	39 s/epoch
memory cost	1520 MB	47624 MB	2050 MB

E. Computing Overhead

We compared the computational overhead of our model with Mamba and Transformer architectures. As shown in TABLE III, our model achieves significantly improved completion performance with computational costs slightly higher than Mamba, but substantially lower than those of the Transformer. This demonstrates our model's efficiency in balancing performance with computational demands.

VII. Conclusion

In this paper, we introduce a novel multi-scale data completion framework designed to address the challenges posed by non-aligned multi-scale data in Sparse Mobile Crowd-Sensing scenarios. We construct multi-scale data under non-aligned compositional relationships using our scale-adjusted resampling architecture and merge it with sparse multi-scale inputs through the SFIM. Then, we employ ST-Mamba and Cross-Mamba for intra-scale and inter-scale interactions, respectively. Our future work delves into the dynamic handling of temporal scales and explores the relationship between multi-scale data collection locations and model performance, aiming to further exploit the potential of multi-scale data.

Acknowledgment

This work is supported in part by Jilin Science and Technology Research Project 20250102220JC, National Natural Science Foundation of China under Grant Nos. 62472194 and 62272193, and Graduate Innovation Fund of Jilin University No. 2025CX212. En Wang is the corresponding author of this paper.

References

[1] L. Wang, D. Zhang, Y. Wang, C. Chen, X. Han, and A. M'hamed, "Sparse mobile crowdsensing: challenges and opportunities," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 161–167, 2016.

[2] F. Calabrese, G. Di Lorenzo, and C. Ratti, "Human mobility prediction based on individual and collective geographical preferences," in *13th International IEEE Conference on Intelligent Transportation systems*, 2010, pp. 312–317.

[3] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, no. 6, pp. 790–808, 2012.

[4] A. Crooks, A. Croitoru, A. Stefanidis, and J. Radzikowski, "# earthquake: Twitter as a distributed sensor system," *Transactions in GIS*, vol. 17, no. 1, pp. 124–147, 2013.

[5] Y. Mostofi, "Compressive cooperative sensing and mapping in mobile networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 12, pp. 1769–1784, 2011.

[6] E. Wang, M. Zhang, Y. Yang, Y. Xu, and J. Wu, "Exploiting outlier value effects in sparse urban crowdsensing," in *IEEE/ACM Ixnternational Symposium on Quality of Service*, 2021, pp. 1–10.

[7] W. Liu, Y. Yang, E. Wang, and J. Wu, "Fine-grained urban prediction via sparse mobile crowdsensing," in *IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems*, 2020, pp. 265–273.

[8] E. Wang, W. Liu, W. Liu, C. Xiang, B. Yang, and Y. Yang, "Spatiotemporal transformer for data inference and long prediction in sparse mobile crowdsensing," in *IEEE Conference on Computer Communications*, 2023, pp. 1–10.

[9] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *IEEE Communications magazine*, vol. 48, no. 9, pp. 140–150, 2010.

[10] S. Liu, H. Yu, C. Liao, J. Li, W. Lin, A. X. Liu, and S. Dustdar, "Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting," in *International Conference on Learning Representations*, 2022, pp. 1–20.

[11] S. Wang, H. Wu, X. Shi, T. Hu, H. Luo, L. Ma, J. Y. Zhang, and J. ZHOU, "Timemixer: Decomposable multiscale mixing for time series forecasting," in *The Twelfth International Conference on Learning Representations*, 2024.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 1–11.

[13] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint:2312.00752*, 2023.

[14] L. Chen, D. Chen, Z. Shang, B. Wu, C. Zheng, B. Wen, and W. Zhang, "Multi-scale adaptive graph neural network for multivariate time series forecasting," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 10, pp. 10 748–10 761, 2023.

[15] J. Zhang, Y. He, W. Chen, L.-D. Kuang, and B. Zheng, "Corrformer: Context-aware tracking with cross-correlation and transformer," *Computers and Electrical Engineering*, vol. 114, p. 109075, 2024.

[16] Y. Liu, Y. Tian, Y. Zhao, H. Yu, L. Xie, Y. Wang, Q. Ye, J. Jiao, and Y. Liu, "VMamba: Visual state space model," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[17] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," in *Forty-first International Conference on Machine Learning*, 2024.

[18] L. Li, H. Wang, W. Zhang, and A. Coster, "Stg-mamba: spatial-temporal graph learning via selective state space model," *arXiv preprint arXiv:2403.12418*, 2024.

[19] H. Wu, K. Wang, F. Xu, Y. Li, X. Wang, W. Wang, H. Wang, and X. Luo, "Spatio-temporal twins with a cache for modeling long-term system dynamics," 2024.

[20] A. Gu, *Modeling Sequences with Structured State Spaces*. Stanford University, 2023.

[21] Z. Wang, F. Kong, S. Feng, M. Wang, X. Yang, H. Zhao, D. Wang, and Y. Zhang, "Is mamba effective for time series forecasting?" *Neurocomputing*, vol. 619, p. 129178, 2025.

[22] A. Nagrani, S. Yang, A. Arnab, A. Jansen, C. Schmid, and C. Sun, "Attention bottlenecks for multimodal fusion," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 200–14 213, 2021.

[23] J. Wei, Z. Li, M. Cribb, W. Huang, W. Xue, L. Sun, J. Guo, Y. Peng, J. Li, A. Lyapustin *et al.*, "Improved 1 km resolution pm 2.5 estimates across china using enhanced space–time extremely randomized trees," *Atmospheric Chemistry and Physics*, vol. 20, no. 6, pp. 3273–3289, 2020.

[24] J. Wei, Z. Li, W. Xue, L. Sun, T. Fan, L. Liu, T. Su, and M. Cribb, "The chinahighpm10 dataset: generation, validation, and spatiotemporal variations from 2015 to 2019 across china," *Environment International*, vol. 146, p. 106290, 2021.

[25] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp. 1655–1661.

[26] Z. Wen, W. Yin, and Y. Zhang, "Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm," *Mathematical Programming Computation*, vol. 4, no. 4, pp. 333–361, 2012.

[27] E. Wang, M. Zhang, X. Cheng, Y. Yang, W. Liu, H. Yu, L. Wang, and J. Zhang, "Deep learning-enabled sparse industrial crowdsensing and prediction," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 9, pp. 6170–6181, 2020.

[28] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[29] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 11 106–11 115.

[30] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," in *The Eleventh International Conference on Learning Representations*, 2023, pp. 1–23.

[31] Z. Shen, M. Zhang, H. Zhao, S. Yi, and H. Li, "Efficient attention: Attention with linear complexities," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 3531–3539.