Urban Aerosol Distribution Prediction System Using Heterogeneous and Uncertain Data Sources

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

M.Sc. Chaofan Li

Tag der mündlichen Prüfung: 21. Juli 2025

1. Referent: Prof. Dr. Michael Beigl
2. Referent: Prof. Dr. Leonidas Ntziachristos

Acknowledgments

To begin with, I wish to convey my sincere gratitude to my advisor, Professor Michael Beigl. Over the past several years, he has provided invaluable guidance, steadfast support, and the autonomy to pursue my research interests. His trust and encouragement have been instrumental in shaping my academic and personal development. I truly appreciate his patience and insightful advice, which have consistently motivated me to broaden my knowledge horizons.

I would also like to extend my heartfelt thanks to my co-advisor, Professor Leonidas Ntziachristos, for offering me a broader perspective beyond the realm of machine learning. His expertise has significantly enhanced my comprehension of urban particulate matter distribution and directed me to explore my research from new angles, which has proven essential in refining my ideas and methodology.

I am indebted to Dr. Till Riedel for his astute insights and constructive discussions, which have been invaluable during my research endeavors. His support in project management has also been profoundly beneficial, facilitating a more structured and efficient approach to complex research processes.

My sincere gratitude also extends to my laboratory colleagues, Dr. Haibin Zhao, Dr. Likun Fang, Dr. Paul Tremper, Dr. Yexu Zhou, and Dr. Yiran Huang, among others. Their experiences, unwavering support, and collaborative spirit have made my research journey productive and enjoyable. Beyond our academic endeavors, the memorable moments we have shared during leisure time have served as significant sources of joy and motivation.

I want to express a special acknowledgment to the administrative staff: Ms. Melissa Alpman, Ms. Zinoula Tsiouma, and Ms. Denise Hillmann. Their dedication and assistance in managing paperwork and administrative affairs have undoubtedly contributed to the smooth progression of my PhD journey. Their support throughout the years has been immensely valuable.

I am also grateful to the organizers and principal investigators of the HEPTA project for offering such an intriguing and meaningful research theme and for their financial support throughout my academic tenure. Furthermore, I would like to convey my appreciation to my fellow PhD candidates involved in the HEPTA project, particularly Dr. Giannis Ioannides and Dr. Panagiotis Pgkirmpas. Working alongside you has been a rewarding experience, and our collaborative efforts have been both productive and memorable.

Finally, I express my profound appreciation to my parents, Mr. Yanhong Li and Mrs. Qingqing Song. Their unwavering support and encouragement have endowed me with the strength and determination to embark on and persevere through this PhD journey. In retrospect, it has been an incredibly enriching experience, and I am sincerely grateful for all they have done for me.

I extend my heartfelt thanks to all who have contributed to this journey!

Abstract

Aerosols are tiny solid or liquid particles suspended in the atmosphere. In urban areas, high concentrations of aerosols are closely associated with respiratory diseases, cardiovascular issues, and overall declines in public health. Given their significant implications, designing a fine-grained urban aerosol distribution prediction system is essential for aiding environmental policymaking, protecting public health, and guiding urban planning.

Observing aerosol distribution depends on two main approaches: remote sensing and in-situ sensor networks. While remote sensing techniques offer extensive spatial coverage, they have significant limitations, such as insufficient spatial resolution, long revisit intervals, and occlusion vulnerability. These limitations render remote sensing unsuitable for real-time and high-resolution monitoring of urban aerosols. As a result, in-situ sensor networks, which provide direct and continuous measurements, continue to be the primary method for observing urban aerosol distribution. However, in-situ sensor networks also encounter challenges despite their benefits, including uneven spatial coverage, data reliability issues, and maintenance constraints.

The core of an urban aerosol distribution prediction system lies in spatiotemporal data analysis models that can be broadly classified into physics-driven and data-driven approaches. Physics-driven models rely on well-established atmospheric transport equations and chemical reaction mechanisms, which makes them interpretable and theoretically robust. However, they often involve expensive calculations during inference, and their effectiveness in fine-grained predictions heavily relies on comprehensive and highly accurate input data, including but not limited to land use, traffic emissions, and various meteorological parameters. Unfortunately, in-situ sensor networks often fail to provide data with the necessary breadth and quality due to technical, financial, and administrative constraints, resulting in models that cannot function

or produce unreliable outputs. Given these limitations, data-driven approaches are increasingly preferred. Unlike physics-driven models, data-driven methods extract statistical patterns from available observations, making them more adaptable to incomplete and imperfect datasets.

Despite the better adaptability of data-driven methods, predicting urban aerosol distribution still encounters significant challenges. The main difficulties arise from the heterogeneity and noise in in-situ sensor network data. Additional factors, such as the unbalanced distributions of features and labels and the incompleteness of patterns, further complicate the task. This dissertation explores various approaches to address these challenges, utilizing imperfect data to develop a more effective spatiotemporal analysis model for urban aerosol distribution prediction.

We explore data augmentation strategies to tackle the challenges of defective and limited data. However, data augmentation methods for urban aerosol distribution are significantly constrained due to the complexity of environmental dynamics. We explore using Computational Fluid Dynamics (CFD) simulations to generate synthetic data. While CFD simulations can yield reliable synthetic aerosol distribution data, their computational costs are prohibitively high, rendering them impractical for supporting the large-scale datasets needed for machine learning models. We, therefore, developed a graph neural network-based CFD surrogate model to address this limitation, significantly accelerating data generation.

Regarding spatiotemporal modeling, we utilize a "divide-and-conquer" strategy to address temporal and spatial correlations separately, reducing misattribution and improving predictive accuracy. We develop a Neural Kernel Network (NKN)-based Gaussian Process Regression (GPR) model to capture temporal correlations. This model leverages GPR to capture long-range temporal dependencies and quantify uncertainty while employing the NKN kernel to overcome the challenge of designing appropriate kernel functions without prior knowledge of the complex underlying system. For spatial correlations, we propose three specialized modules to tackle distinct challenges. The Context Encoder Spatial Interpolation (CESI) model is crafted to manage heterogeneous sensor data by effectively mining sparse inputs. The Information Segmentation

Spatial Interpolation (ISSI) model utilizes self-supervised learning to alleviate uncertainties that arise from unobserved Latent Context Information in sensor networks. Lastly, the Feature Deviation Embedding Graph Spatial Interpolation (FDE-GSI) model addresses feature imbalance by implementing Feature Deviation Embedding and an adaptive information bottleneck mechanism.

We evaluate our proposed models using multiple real-world datasets, demonstrating their accuracy, robustness, and adaptability. Unlike existing models that show significant performance fluctuations across datasets, our models consistently achieve state-of-the-art performance. This stability highlights the effectiveness of our design choices and the potential for application deployments.

In summary, this dissertation tackles the critical challenge of predicting fine-grained urban aerosol distribution using heterogeneous and uncertain data sources. By systematically analyzing the limitations of existing observational methods and prediction frameworks, we propose a series of innovations, including CFD-based data augmentation, GNN-based CFD surrogate modeling, and a suite of specialized temporal and spatial analysis models. Our models effectively address key challenges such as data heterogeneity, uncertainty, and feature imbalance, achieving state-of-the-art performance across multiple real-world datasets. This work advances the field of urban environmental modeling, providing scalable and robust solutions for aerosol distribution prediction in complex urban environments.

Zusammenfassung

Aerosole sind winzige feste oder flüssige Partikel, die in der Atmosphäre suspendiert sind. In städtischen Gebieten sind hohe Aerosolkonzentrationen eng mit Atemwegserkrankungen, Herz-Kreislauf-Problemen und einem allgemeinen Rückgang der öffentlichen Gesundheit verbunden. Angesichts ihrer erheblichen Auswirkungen ist die Entwicklung eines hochauflösenden Vorhersagesystems für die städtische Aerosolverteilung entscheidend, um umweltpolitische Entscheidungen zu unterstützen, die öffentliche Gesundheit zu schützen und die Stadtplanung zu optimieren.

Die Beobachtung der Aerosolverteilung basiert auf zwei Hauptansätzen: Fernerkundung und in-situ Sensornetzwerke. Während Fernerkundungstechniken eine großflächige Abdeckung ermöglichen, weisen sie erhebliche Einschränkungen auf, wie z. B. eine unzureichende räumliche Auflösung, lange Wiederholungsintervalle und eine Anfälligkeit für Hindernis. Diese Einschränkungen machen die Fernerkundung für die Echtzeit- und hochauflösende Überwachung städtischer Aerosole ungeeignet. Infolgedessen bleiben in-situ Sensornetzwerke, die direkte und kontinuierliche Messungen liefern, die primäre Methode zur Beobachtung der städtischen Aerosolverteilung. Trotz ihrer Vorteile stehen jedoch auch in-situ Sensornetzwerke vor Herausforderungen, darunter eine ungleichmäßige räumliche Abdeckung, Probleme mit der Datenzuverlässigkeit und Wartungsaufwand.

Das Herzstück eines Vorhersagesystems für die städtische Aerosolverteilung liegt in spatiotemporalen Datenanalysemodellen, die grob in physikgetriebene und datengetriebene Ansätze unterteilt werden können. Physikgetriebene Modelle basieren auf etablierten atmosphärischen Transportgleichungen und chemischen Reaktionsmechanismen, was sie interpretierbar und theoretisch robust macht. Allerdings erfordern sie oft rechenaufwendige Berechnungen während der Inferenz, und ihre Genauigkeit in hochauflösenden Vorhersagen hängt stark

von umfassenden und hochpräzisen Eingabedaten ab, darunter Landnutzung, Verkehrsemissionen und verschiedene meteorologische Parameter. Aufgrund technischer, finanzieller und administrativer Einschränkungen liefern in-situ Sensornetzwerke jedoch oft nicht die erforderliche Datenqualität und -abdeckung, was dazu führt, dass physikgetriebene Modelle entweder nicht funktionieren oder unzuverlässige Ergebnisse liefern. Angesichts dieser Einschränkungen gewinnen datengetriebene Ansätze zunehmend an Bedeutung. Im Gegensatz zu physikgetriebenen Modellen extrahieren datengetriebene Methoden statistische Muster aus vorhandenen Beobachtungen und sind dadurch anpassungsfähiger an unvollständige und ungenaue Datensätze.

Trotz ihrer besseren Anpassungsfähigkeit stehen datengetriebene Methoden zur Vorhersage der städtischen Aerosolverteilung weiterhin vor erheblichen Herausforderungen. Die Hauptprobleme resultieren aus der Heterogenität und dem Rauschen in den Daten der in-situ Sensornetzwerke. Weitere Faktoren, wie unausgewogene Merkmals- und Labelverteilungen sowie unvollständige Muster, erschweren die Aufgabe zusätzlich. Diese Dissertation untersucht verschiedene Ansätze zur Bewältigung dieser Herausforderungen, indem sie unvollständige Daten nutzt, um ein effektiveres spatiotemporales Analysemodell zur Vorhersage der städtischen Aerosolverteilung zu entwickeln.

Wir untersuchen Datenaugmentierungsstrategien, um die Herausforderungen defekter und begrenzter Daten zu bewältigen. Allerdings sind Datenaugmentierungsmethoden für die städtische Aerosolverteilung aufgrund der Komplexität der Umweltdynamik stark eingeschränkt. Daher untersuchen wir die Verwendung von Computational Fluid Dynamics (CFD)-Simulationen zur Generierung synthetischer Daten. Obwohl CFD-Simulationen zuverlässige synthetische Aerosolverteilungsdaten liefern können, sind ihre Rechenkosten extrem hoch, was sie für die Unterstützung großflächiger Datensätze, die für maschinelle Lernmodelle erforderlich sind, unpraktikabel macht. Um dieses Problem zu lösen, haben wir ein Graph Neural Network (GNN)-basiertes CFD-Surrogatmodell entwickelt, das die Datengenerierung erheblich beschleunigt.

Im Bereich der spatiotemporalen Modellierung verwenden wir eine "Divide-and-Conquer"-Strategie, um zeitliche und räumliche Korrelationen getrennt zu analysieren, Fehlzuweisungen zu reduzieren und die Vorhersagegenauigkeit

zu verbessern. Zur Modellierung zeitlicher Korrelationen entwickeln wir ein Neural Kernel Network (NKN)-basiertes Gaussian Process Regression (GPR)-Modell. Dieses Modell nutzt GPR, um langfristige zeitliche Abhängigkeiten zu erfassen und Unsicherheiten zu quantifizieren, während der NKN-Kernel das Problem der manuellen Definition geeigneter Kernelfunktionen für komplexe Systeme überwindet. Zur Modellierung räumlicher Korrelationen schlagen wir drei spezialisierte Module vor, die unterschiedliche Herausforderungen bewältigen. Das Context Encoder Spatial Interpolation (CESI)-Modell wurde entwickelt, um heterogene Sensordaten durch effektives Mining spärlicher Eingaben zu verarbeiten. Das Information Segmentation Spatial Interpolation (ISSI)-Modell nutzt selbstüberwachtes Lernen, um Unsicherheiten zu verringern, die aus nicht beobachteter Latent Context Information in Sensornetzwerken entstehen. Schließlich adressiert das Feature Deviation Embedding Graph Spatial Interpolation (FDE-GSI)-Modell Merkmalsungleichgewichte, indem es Feature Deviation Embedding und einen adaptiven Informations-Engpass-Mechanismus implementiert.

Unsere vorgeschlagenen Modelle werden anhand mehrerer realer Datensätze evaluiert, wobei sie ihre Genauigkeit, Robustheit und Anpassungsfähigkeit unter Beweis stellen. Im Gegensatz zu bestehenden Modellen, die starke Leistungsfluktuationen über verschiedene Datensätze hinweg zeigen, erreichen unsere Modelle durchgängig State-of-the-Art-Leistungen. Diese Stabilität unterstreicht die Wirksamkeit unserer Designentscheidungen und ihr Potenzial für praktische Anwendungen.

Zusammenfassend befasst sich diese Dissertation mit der Herausforderung, die hochauflösende städtische Aerosolverteilung mithilfe heterogener und unsicherer Datenquellen vorherzusagen. Durch eine systematische Analyse der Einschränkungen bestehender Beobachtungsmethoden und Vorhersagemodelle schlagen wir eine Reihe von Innovationen vor, darunter CFD-basierte Datenaugmentierung, GNN-basierte CFD-Surrogatmodellierung und spezialisierte spatiotemporale Analysemodelle. Unsere Modelle bewältigen zentrale Herausforderungen wie Datenheterogenität, Unsicherheit und Merkmalsungleichgewicht und erreichen State-of-the-Art-Leistungen auf mehreren realen Datensätzen. Diese Arbeit leistet einen bedeutenden Beitrag zur städtischen Umweltmodellierung und bietet skalierbare und robuste Lösungen zur Aerosolverteilungsvorhersage in komplexen städtischen Umgebungen.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

Aerosols are suspensions of tiny solid or liquid particles in the atmosphere, originating from both natural and anthropogenic sources. Natural sources include volcanic eruptions, sea spray, and wildfires, while human activities such as fossil fuel combustion, industrial emissions, and vehicular exhaust contribute significantly to aerosol concentrations in urban areas. Due to their small size (often in the micrometer or sub-micrometer range), aerosols can remain airborne for extended periods, facilitating their transport over vast distances and penetrating deep into the human respiratory system. Their presence in urban environments poses serious threats to public health, as prolonged exposure to high aerosol concentrations has been linked to respiratory diseases, cardiovascular disorders, and increased mortality rates [2; 18; 29; 81; 87; 99]. Additionally, they contribute to visibility reduction, urban haze, and environmental degradation [53], further impacting daily life and economic activities. Given their far-reaching consequences, accurately predicting urban aerosol distribution is critical for safeguarding public health, formulating effective environmental policies, and guiding sustainable urban planning.

Nowadays, monitoring aerosol distribution relies primarily on remote sensing and in-situ sensor networks. Remote sensing techniques, such as satellite-based and ground-based lidar observations, provide large-scale spatial coverage. However, current remote sensing techniques for aerosol monitoring have significant limitations, including low spatial resolution, long revisit intervals, and susceptibility to occlusions such as clouds and different aerosol height profiles [71; 128]. These drawbacks make remote sensing unsuitable for real-time, high-resolution monitoring of urban aerosol distribution.

In contrast, in-situ sensor networks provide direct and continuous measurements of aerosol concentrations at ground level. These networks consist of fixed monitoring stations and mobile sensor units that record fine-grained aerosol variations in urban environments. However, they also face challenges such as uneven spatial coverage, data reliability issues, and high maintenance costs [20; 45]. Despite these constraints, in-situ sensor networks remain the most popular approach for monitoring urban aerosol distribution.

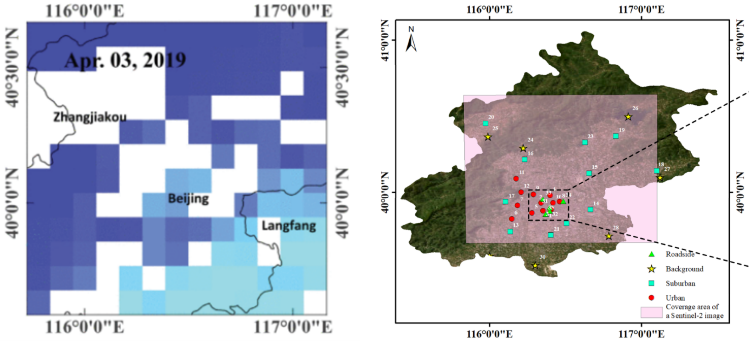


Figure 1.1: An illustrative example demonstrating the differences in aerosol observation methods for the same area, images from [137]. (Left) Remote sensing provides wide spatial coverage but suffers from low spatial resolution, long revisit intervals, and missing data due to occlusions. (Right) In-situ sensor networks offer temporal continuous ground level observations. However, only at sensor deployment locations.

At the heart of an urban aerosol distribution prediction system lies a spatiotemporal analysis model that captures the complex spatial and temporal variations in aerosol concentrations. Since in-situ sensor networks currently serve as the primary method for monitoring urban aerosol distribution, the choice of modeling approach must account for their unique characteristics and constraints. This ensures the system can deliver accurate and reliable predictions in real-world urban environments.

1.2 Problem and Motivation

Spatiotemporal analysis models for urban aerosol distribution prediction can be classified into physics-driven and data-driven approaches. Physics-driven models simulate aerosol dispersion using well-established atmospheric transport equations and chemical reaction mechanisms. These models, such as Computational Fluid Dynamics (CFD) simulations [89] and Chemical Transport Models (CTMs) [40], provide strong theoretical foundations and interpretability, making them highly valuable for studying pollutant dynamics under various environmental conditions. However, their effectiveness heavily depends on the availability of comprehensive and highly accurate input data, including meteorological conditions, emission inventories, and land use information [21]. Unfortunately, providing complete and precise input data in real-world applications is often infeasible due to technical, financial, and administrative constraints. Moreover, physics-driven models involve high computational costs during inference, making real-time applications impractical even with high-performance computing resources.

In contrast, data-driven models leverage machine learning and statistical methods to infer aerosol distribution patterns directly from observed data. Unlike physics-driven models, they do not require explicit knowledge of the underlying physical processes and are less reliant on the availability of specific types of input data. Instead, they adaptively extract relevant patterns from available observations and use them for prediction. This adaptability makes data-driven approaches more robust to incomplete and imperfect datasets, a key advantage given the limitations of current observation techniques. Additionally, data-driven models typically have lower computational costs during inference, making them more suitable for practical deployment. As a result, many spatiotemporal analysis models based on data-driven approaches have been developed in recent years [31; 80; 142].

However, through extensive evaluation of existing data-driven spatiotemporal analysis models on real-world urban aerosol sensor network datasets, we find that their performance is often volatile. Because these models rely on statistical correlations extracted from datasets, their effectiveness is highly sensitive to the quality and representativeness of the data. Unfortunately, real-world urban aerosol sensor networks, designed to maximize spatial coverage under limited budgets, often employ low-cost sensors and varying sensor deployment densities across different locations (thereafter referred to as hybrid sensor networks). These factors introduce significant variability and uncertainty in data quality, leading to challenges in model generalization [27]. Chapter 3 presents a detailed case study analyzing a real-world hybrid sensor network dataset, summarizing its typical challenges, including high heterogeneity, high uncertainty, and feature imbalance.

Without tailored solutions to address these challenges, data-driven models struggle to generalize beyond their training datasets, creating a significant gap to real-world deployment.

1.3 Aim and Objectives

This research aims to bridge the gap between the limitations of hybrid sensor network datasets and the existing spatiotemporal analysis methods in urban aerosol distribution prediction. Due to the inherent issues in real-world sensor network datasets, conventional data-driven models often struggle with generalization, robustness, and reliability. To address these challenges, this dissertation focuses on designing data-driven modeling strategies that can effectively leverage imperfect and incomplete data while ensuring accurate and stable aerosol distribution predictions.

To achieve this aim, the dissertation sets out the following key objectives:

1. **Identification of key challenges in hybrid sensor network datasets**: Conduct a thorough analysis of real-world urban aerosol sensor network datasets to identify critical challenges. This analysis will serve as the foundation for designing robust data-driven solutions.

2. **Development of tailored solutions across the entire data-driven modeling pipeline**: Design and implement methodologies that address the identified challenges across different stages of the data-driven modeling pipeline. This includes data preprocessing, feature extraction, model architecture design, and uncertainty quantification, ensuring that the developed methods can effectively handle noisy and incomplete sensor data.

3. **Comprehensive Evaluation Framework**: Establish a rigorous evaluation framework to systematically assess the proposed solutions regarding prediction accuracy, robustness, and adaptability. The evaluation will be conducted on multiple real-world sensor network datasets to ensure the generalizability and practical feasibility of the developed methods.

By accomplishing these objectives, this dissertation aims to advance the field of urban aerosol distribution modeling. It provides innovative and scalable solutions that enhance the practical usability of data-driven models in real-world environmental monitoring applications.

1.4 Contributions

Regarding the above aims and objectives, this dissertation presents a series of contributions that provide solutions for urban aerosol distribution modeling.

1.4.1 Data Augmentation

Data augmentation is a commonly used technique to address dataset limitations, but its application in urban aerosol distribution modeling is highly constrained. Unlike other domains, where simple heuristics such as scaling, translation, flipping, and rotation are practical, these methods can be hazardous in geospatial data, where latitude, longitude, and azimuth are crucial. Additionally, due to the complexity of urban atmospheric dynamics, directly synthesizing realistic aerosol distribution data is a non-trivial challenge. The influence of meteorological conditions, local emissions, and urban structures makes it difficult to generate plausible artificial data without violating fundamental physical principles.

> **Question 1**: CFD simulations have been considered a promising approach for generating realistic synthetic data due to their ability to model airflow and pollutant dispersion in urban environments. Can CFD simulations be leveraged to create artificial data that closely approximates real-world urban aerosol distributions?

To reduce the dependency of CFD models on highly accurate and complex input data, we designed a CFD simulation framework that primarily relies on easily obtainable urban and meteorological data. To validate the effectiveness of this CFD-based urban aerosol distribution model, we conducted extensive data collection and executed simulations within a constrained time window. The generated aerosol distribution patterns were then compared against real-world recorded observations. Our results indicate that while discrepancies remain, the generated synthetic data closely aligns with real-world observations.

> **Contribution 1**: We propose a CFD-based urban aerosol distribution simulation model capable of generating realistic synthetic aerosol data. While this model has been validated only within a limited temporal window, and its computational efficiency and precision are insufficient for direct predictive modeling, it is already enough for data augmentation. The synthetic data generated by our approach can balance feature distributions, supplement missing patterns, and provide a more comprehensive training foundation for data-driven models. This contribution helps bridge data gaps, enabling better generalization and robustness in urban aerosol prediction systems.

CFD simulations are well known for their high computational cost during inference, making them impractical for large-scale data generation. However, data-driven models require vast training data, meaning that even if CFD is only used to supplement a small number of critical patterns, improving its computational efficiency remains essential. The challenge lies in balancing the physical accuracy of CFD simulations with the scalability needed for data augmentation in urban aerosol distribution modeling.

> **Question 2**: To address this challenge, we explore the feasibility of designing a data-driven surrogate model for CFD simulations, capable of approximating CFD outputs with significantly lower inference time. Can a machine learning-based surrogate model be trained on a limited number of CFD simulations while achieving high accuracy and improv-

> ing inference speed?

Graph Neural Networks (GNNs) have been recognized as a promising approach for modeling unstructured mesh-based CFD simulations, as they naturally capture relationships between spatially interconnected points. However, existing GNN methods suffer from inefficient message passing and oversmoothing, leading to degraded performance in iterative simulations. To address these limitations, we design a graph-structured learning-based CFD surrogate model, which dynamically adjusts the graph topology based on the current state. This adaptive graph structure improves prediction accuracy while mitigating error accumulation over multiple time-step iterations. We evaluate our approach in various datasets of varying scales, demonstrating its robustness across diverse application settings.

> **Contribution 2**: We propose a graph-structured learning-based CFD surrogate model that dynamically adjusts graph structures based on the evolving states. This approach enhances prediction accuracy and reduces error accumulation in multi-step inference. Most importantly, as a machine learning-based surrogate model, it significantly reduces the computational cost of CFD inference, making CFD-driven data augmentation more feasible for urban aerosol distribution modeling. By leveraging GNNs to approximate CFD behavior, this work paves the way for efficient large-scale simulations that were previously impractical due to computational constraints.

1.4.2 Spatiotemporal Analysis

The limitations of hybrid sensor network datasets primarily stem from the widespread use of low-cost sensors. To achieve affordability, these sensors compromise accuracy and stability, making them inherently prone to higher noise levels, calibration drift, and reduced reliability. In extreme cases, certain low-cost sensors can only perform qualitative measurements, further complicating the task of precise aerosol distribution modeling. As a result, the role of low-cost sensors in urban aerosol monitoring remains controversial, with concerns that their introduction might harm more than it could benefit.

> **Question 3**: A persistent criticism in the field is that the challenges introduced by low-cost sensors outweigh their benefits. The uncertainty and inconsistencies they introduce could, in theory, undermine model reliability rather than enhance predictive power. This raises a critical question: Are low-cost sensors more of a burden than an asset in hybrid sensor networks?

To evaluate this criticism, we conducted comparative experiments while designing the Neural Kernel Network Deep Kernel Learning (NKNDKL) model. Specifically, we assessed model performance across two datasets: The full hybrid sensor network dataset, which included high-precision reference monitors and low-cost sensors, and the filtered dataset, which excludes all low-cost sensors. By analyzing the performance gap between these two settings, we could quantify the contribution of low-cost sensors. Our results reveal that despite the analytical challenges posed by low-cost sensors, their inclusion still significantly improves model performance when proper data-driven methodologies are employed. Despite their inherent inaccuracies, low-cost sensors provide additional spatial and temporal information, enabling more robust and fine-grained urban aerosol predictions when adequately processed.

> **Contribution 3**: Through rigorous experimentation, we demonstrate that the benefits of integrating low-cost sensors far outweigh the challenges they introduce. This finding establishes the necessity of designing spatiotemporal analysis methods that explicitly address and compensate for the limitations of low-cost sensors. Instead of discarding their data due to noise and uncertainty, effective analytical models can harness their contributions to enhance predictive accuracy, making urban aerosol monitoring more scalable, accessible, and informative.

Through extensive analysis, we identified several critical challenges inherent in hybrid sensor network datasets, significantly impacting the performance of data-driven urban aerosol prediction models. These challenges include but are not limited to:

- **Heterogeneity:** Sensor data originates from different sources, including high-cost reference monitors and low-cost sensors with varying degrees of precision. This leads to inconsistencies in measurement distributions and biases across various locations.

- **Uncertainty:** Due to sensor drift, environmental noise, and missing data, predictions must be made under incomplete and unreliable observations, requiring models to be robust against noisy inputs.

- **Feature and Label Imbalance:** In real-world sensor deployments, certain areas are densely monitored while others lack sufficient coverage, leading to imbalanced spatial feature distributions and unevenly sampled ground-truth labels.

These factors pose significant challenges for training accurate and robust spatiotemporal analysis models.

> **Question 4**: What methodological strategies can be employed to train precise and robust spatiotemporal analysis models on hybrid sensor network datasets with inherent flaws?

To address this problem, we systematically explored various advanced learning strategies, such as self-supervised learning, contrastive learning, and adaptive information bottleneck mechanisms. One key finding was that designing appropriate inductive bias constraints is crucial for achieving stable performance on flawed datasets. We observed that self-supervised and adversarial constraints improved the model's adaptability across different tasks. Extensive experimental results confirm that our proposed approaches significantly enhance the accuracy and stability of spatiotemporal analysis models on challenging hybrid sensor network datasets.

> **Contribution 4**: By summarizing the methodological principles behind our successful approaches, we identify a fundamental insight: Balanc-

> ing model adaptability and inductive bias constraints is key to designing effective data-driven models on flawed datasets. Adaptability enables the model to capture fine-grained correlations, which are crucial for improving predictive performance. However, hybrid sensor network datasets are full of misleading patterns, and excessive adaptability can cause overfitting to spurious correlations, leading to erroneous attributions and poor generalization. Therefore, proper inductive bias constraints must be incorporated into the model design. This provides a conceptual framework for developing spatiotemporal models for urban aerosol prediction on hybrid sensor network datasets.

1.5 Structure of This Thesis

This dissertation is divided into seven chapters, each exploring different aspects of developing an urban aerosol prediction system. Figure 1.2 illustrates the thesis structure, providing an overview of the relationships between chapters and subsections. This helps readers quickly understand how the various components interconnect and collectively contribute to the overarching goal of designing an effective urban aerosol prediction system.

Chapter 2 formalizes the definition of spatiotemporal analysis models and outlines the methodological framework adopted in this research. The insights derived from this analysis lay the foundation for the subsequent development of novel methods in later chapters. Additionally, the experimental setup used throughout the dissertation, including datasets, data preparation methods, model training configurations, and evaluation metrics, is detailed in this chapter to avoid redundancy in later discussions.

Chapters 3 to 6 follow the typical machine learning (ML) workflow, covering understanding the data, data preparation, model design, and model evaluation.

Chapter 3 presents SmartAQnet, a real-world urban air pollution monitoring project, as a case study. This chapter provides an in-depth analysis of its

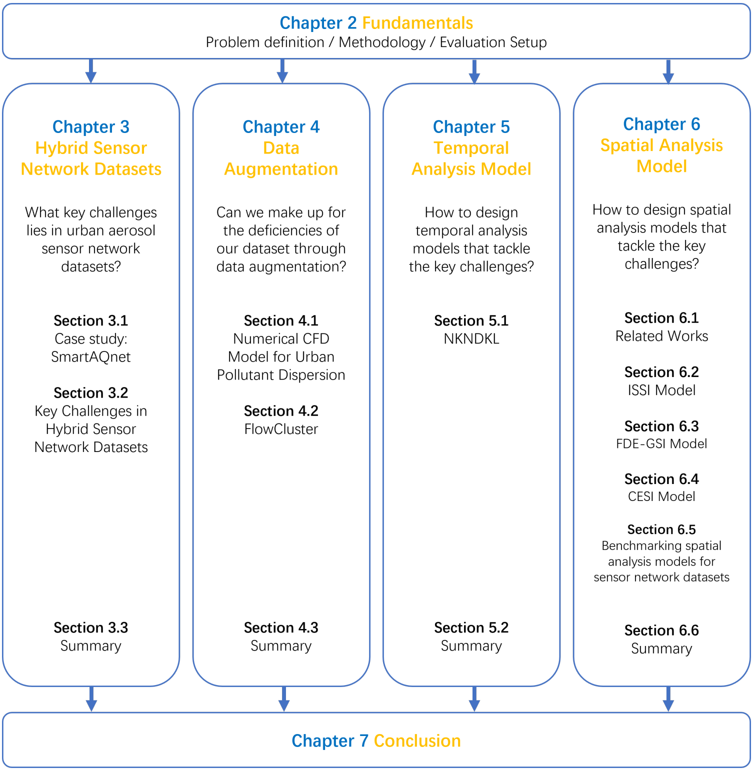

Chapter 2 Fundamentals
Problem definition / Methodology / Evaluation Setup
Chapter 3
Hybrid Sensor Network Datasets
What key challenges lies in urban aerosol sensor network datasets?
Section 3.1
Case study: SmartAQnet
Section 3.2
Key Challenges in Hybrid Sensor Network Datasets
Section 3.3
Summary
Chapter 4
Data Augmentation
Can we make up for the deficiencies of our dataset through data augmentation?
Section 4.1
Numerical CFD Model for Urban Pollutant Dispersion
Section 4.2
FlowCluster
Section 4.3
Summary
Chapter 5
Temporal Analysis Model
How to design temporal analysis models that tackle the key challenges?
Section 5.1
NKNDKL
Section 5.2
Summary
Chapter 6
Spatial Analysis Model
How to design spatial analysis models that tackle the key challenges?
Section 6.1
Related Works
Section 6.2
ISSI Model
Section 6.3
FDE-GSI Model
Section 6.4
CESI Model
Section 6.5
Benchmarking spatial analysis models for sensor network datasets
Section 6.6
Summary
Chapter 7 Conclusion

Figure 1.2: The structure of the dissertation focuses on urban aerosol distribution prediction tasks, with Chapter 1 (Introduction) excluded. Abbreviations: NKNDKL stands for the Neural Kernel Network Deep Kernel Learning Model, ISSI stands for the Information Segmentation Spatial Interpolation Model, FDE-GSI stands for the Feature Deviation Embedding Graph Spatial Interpolation Model, and CESI stands for the Context Encoder Spatial Interpolation Model.

structure, uncovering and explaining the inherent challenges associated with urban aerosol monitoring datasets.

Chapter 4 explores data augmentation techniques to mitigate dataset limitations. Specifically, it introduces two key contributions: (1) synthetic data generation using CFD-based simulations and (2) a Graph Structure Learning (GSL) based surrogate model to enhance the computational efficiency of CFD simulations.

Chapter 5 discusses our attempts to model temporal correlations within urban aerosol monitoring datasets. It introduces the Neural Kernel Network Deep Kernel Learning Model in detail.

Chapter 6 focuses on modeling spatial correlations within urban aerosol monitoring datasets. It introduces three novel approaches: the Information Segmentation Spatial Interpolation (ISSI) Model, the Feature Deviation Embedding Graph Spatial Interpolation (FDE-GSI) Model, and the Context Encoder Spatial Interpolation (CESI) Model. These models address challenges such as data heterogeneity, feature imbalance, and missing observations, improving the robustness of aerosol distribution predictions. This chapter also introduces a benchmarking pipeline for spatial analysis models that analyze hybrid sensor network datasets.

Chapter 7 summarizes the key findings of this research and discusses potential future directions for advancing urban aerosol prediction systems.

1.6 List of Publications

The following list gives a comprehensive overview of all scientific papers published by the author that are relevant to this dissertation. Significant parts of this dissertation (across all chapters) were copied from the relevant papers listed below and assembled into a coherent monograph structure.

Chaofan Li, Matthias Budde, Paul Tremper, Klaus Schäfer, Johannes Riesterer, Johanna Redelstein, Erik Petersen, Mohamed Khedr, Xiangsheng Liu, Marcel Köpke, Sajjad Hussain, Felix Ernst, Michal Kowalski, Markus Pesch, Johannes Werhahn, Markus Hank, Andreas Philipp, Josef Cyrys, Jürgen Schnelle-Kreis, Hans Grimm, Volker Ziegler, Annete Peters, Stefan Emeis, Till Riedel, Michael Beigl. "Smartaqnet 2020: a new open urban air quality dataset from heterogeneous pm sensors". In: *Proscience 8 (2022)*

Chaofan Li, Till Riedel, Michael Beigl. "Neural Kernel Network Deep Kernel Learning for Predicting Particulate Matter from Heterogeneous Sensors with Uncertainty". In: *International Conference on Information Integration and Web 2022 (pp. 252-266).*

Yiran Huang, **Chaofan Li**, Hansen Lu, Till Riedel, Michael Beigl. "State graph based explanation approach for black-box time series model". In: *World Conference on Explainable Artificial Intelligence (XAI) 2023 (pp. 153-164)*

Giannis Ioannidis, **Chaofan Li**, Paul Tremper, Till Riedel and Leonidas Ntziachristos. "Application of CFD Modelling for Pollutant Dispersion at an Urban Traffic Hotspot". In: *Atmosphere 15.1 (2024): 113.*

Chaofan Li, Till Riedel, Michael Beigl. "Feature Deviation Embedding Improves Graph Structure Learning for Spatial Interpolation". In: *SIAM International Conference on Data Mining (SDM) 2025*

Chaofan Li, Till Riedel, Michael Beigl. "Isolating Latent Context Information Enhances Graph Structure Learning for Spatial Interpolation". In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) 2025*

2 Fundamentals

2.1 Problem Definition

In this section, we formally define concepts, notations, and the spatiotemporal prediction task.

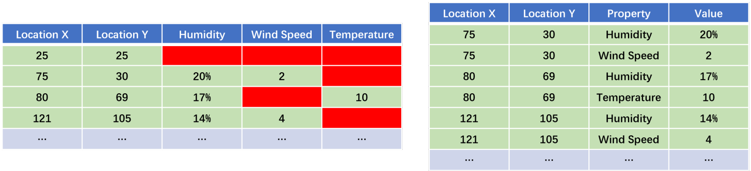
Observed Property: We use the term "Observed Property" from the OGC SensorThings API [70] to refer to the phenomenon of an Observation, abbreviated as OP. For example, in the first row of the data entry in Figure 2.1 (right), the humidity at location (75, 30) is 20%. Here, "humidity" is the OP corresponding to this Observation. Since the spatial distribution of the OP we want to perform spatial interpolation is usually not only affected by spatial correlation but also correlated with some other OPs, there are typically multiple OPs in hybrid WSN datasets. In our input data, these OPs are encoded as different one-hot vectors.

Target Observed Property: We refer to the Observed Property that we want to perform spatiotemporal prediction as the Target Observed Property, abbreviated as OP_{tgt}. For each spatiotemporal prediction model, we only consider one OP_{tgt}.

Support Observed Property: We refer to all other Observed Properties other than OP_{tgt} in the dataset as Support Observed Properties, abbreviated as OP_{sup}. OP_{sup} may also affect the distribution of OP_{tgt}. For example, the distribution of air pollutants is affected by wind speed and wind direction, so hybrid WSNs that measure air pollutants may also deploy sensors that observe these two OP_{sup}. The spatiotemporal prediction model must also be able to learn how these OP_{sup} affect the distribution of OP_{tgt}.

Observation: The reading value of a specific Observed Property at a particular location and time is called an Observation, abbreviated as $t = (OP, C, V, t)$. C is the spatial coordinate corresponding to the Observation, which might be a two- or three-dimensional vector. V is a real number indicating the reading value corresponding to the Observation. t is the time corresponding to the Observation.

Observation Frame: We define the sequence of all Observations in the same period as an Observation Frame $F = \{t_1, t_2, ..., t_n\}$. Obviously, the Observation Frame is a narrow-format data table like it is shown in Figure 2.1 (right). An Observation Frame can be further divided into two parts: Target OP Sequence F_{tgt} refers to the Observations of the Target OP, and Support OP Sequence F_{sup} refers to the Observations of Support OPs.



Location X	Location Y	Humidity	Wind Speed	Temperature
25	25	🟥	🟥	🟥
75	30	20%	2	🟥
80	69	17%	🟥	10
121	105	14%	4	🟥
...	...	...	...	...

Location X	Location Y	Property	Value
75	30	Humidity	20%
75	30	Wind Speed	2
80	69	Humidity	17%
80	69	Temperature	10
121	105	Humidity	14%
121	105	Wind Speed	4
...	...	...	...

Figure 2.1: An example of the wide format (left) and narrow format (right) of the same Observation Frame of hybrid sensor network datasets.

Spatiotemporal prediction Task: Given a dataset $D = \{F_1, F_2, ..., F_n\}$ containing all historical Observation Frames of a hybrid WSN. The spatiotemporal prediction task is to predict the value V' of the target OP in a new Observation Frame F' (F' may not in D) at any arbitrary target location C' and time T'. The basis for prediction comes from the spatial correlation with the known values in $F^{'}_{tgt}$ and the effect of $F^{'}_{sup}$ on this correlation, which can be learned from historical Observation Frames provided in D, and the temporal correlation between all historical Observation Frames in D and the new Observation Frame F'.

2.2 Discussion on Technical Approaches

When designing spatiotemporal analysis models, two primary strategies can be employed: a direct approach, where a single module simultaneously learns both temporal and spatial correlations, and a divide-and-conquer approach, where separate modules handle temporal and spatial dependencies independently. While the direct approach has notable advantages, particularly in handling heterogeneous datasets, its susceptibility to erroneous attributions makes it less suitable for flawed sensor network data. In contrast, the divide-and-conquer approach provides greater robustness in such conditions by mitigating the risks associated with data imperfections.

The direct approach, which learns spatial and temporal relationships jointly, is often preferred for heterogeneous datasets with changing sensor distributions. Its primary strength lies in its flexibility in capturing complex cross-domain dependencies, as spatial and temporal information are learned in an integrated feature space. This ability makes the direct approach particularly effective in handling sensor heterogeneity, where different sensor types or locations may exhibit distinct patterns. By jointly optimizing spatial and temporal representations, the model can automatically adjust for inconsistencies across different regions, making it more adaptable to non-uniform data distributions.

However, this same flexibility becomes a major weakness when dealing with flawed hybrid sensor network datasets that contain inherent noise, missing values, and biased observations. The primary issue arises from erroneous attributions, where the model, instead of learning genuine spatiotemporal relationships, learns spurious correlations introduced by sensor inaccuracies, unobserved influencing factors, or imbalanced feature distributions. Because the direct approach entangles spatial and temporal learning, it lacks explicit mechanisms to disentangle misleading correlations from true causal relationships. This problem is exacerbated in flawed hybrid sensor network datasets where pattern missing or low-quality sensor readings create gaps in the feature space, forcing the model to interpolate based on incomplete or biased data. As a result, the direct approach can reinforce false dependencies, leading to overfitting on spurious patterns rather than learning meaningful environmental trends.

The divide-and-conquer approach, by contrast, explicitly separates temporal and spatial learning into distinct modules, significantly reducing the risks associated with erroneous attributions. By handling temporal dependencies independently, the model ensures that time-series patterns are learned without being overly influenced by spatial inconsistencies. Likewise, the spatial module focuses purely on geospatial interpolation and correlation extraction,

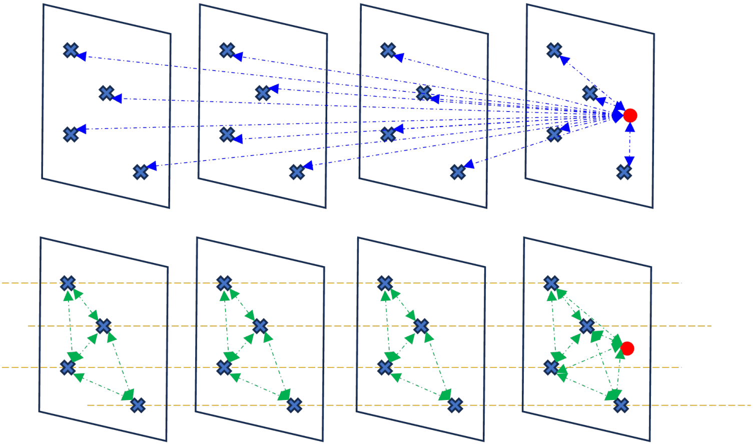


Figure 2.2: (Top) Direct approach: A single module jointly analyzes temporal and spatial correlations between observations (blue dashed arrows). (Bottom) Divide-and-conquer approach: The spatial analysis module captures spatial correlations among observations within the same time frame (green dashed arrows), while the temporal analysis module learns temporal correlations across observation frames (orange dashed lines).

preventing short-term fluctuations or sensor-specific biases from distorting its feature representations. This structured decomposition not only enhances interpretability but also introduces an implicit form of regularization, where each module is optimized for its specific subtask rather than simultaneously trying to infer multiple relationships from imperfect data.

More importantly, the divide-and-conquer approach is naturally more robust to flawed datasets, as it provides explicit control over error propagation between spatial and temporal components. In hybrid sensor networks where low-cost sensors contribute varying levels of noise and accuracy, this separation prevents one domain from contaminating the learning process of the other. For example, if a sensor reports unreliable readings at irregular intervals, a direct model may incorrectly propagate its influence across both spatial and temporal dimensions, leading to distortions in the learned aerosol distribution patterns. In contrast, a divided approach ensures that spatial inconsistencies do not directly interfere with the model's ability to extract meaningful temporal trends, and vice versa.

Given these factors, while direct models remain useful in controlled, high-quality datasets, the divide-and-conquer approach emerges as the superior choice when working with real-world, flawed hybrid sensor network data. By systematically managing uncertainty and preventing cross-domain contamination, it provides a more stable, interpretable, and generalizable framework for urban aerosol prediction.

2.3 Experimental Setup

2.3.1 Hardware and Software Setup

We conduct our experiments on an HPC cluster. Models that demand lower computational resources are trained on CPU nodes equipped with 20 Intel Xeon Gold 6230 CPUs and 192 GB of memory. Other models are trained on GPU nodes equipped with 20 Intel Xeon Gold 6230 CPUs, 192 GB of CPU memory, 2 NVIDIA Tesla V100 GPUs, and 64 GB of GPU memory.

The system used for all nodes is Red Hat Enterprise Linux (RHEL) 8.4. The training environment is based on Python 3.10.12, Pytorch 2.1.0 + CUDA 12.1, Pytorch-geometric 2.4.0, DGL 2.2.1, Numpy 1.26.1, Pandas 2.1.1, Scikit-learn 1.3.1, and Scipy 1.11.3.

2.3.2 Experimental Setup for GNN-based CFD Surrogate Models

2.3.2.1 Datasets. For the CFD surrogate model, all experiments are conducted on three publicly available datasets: the Airfoil dataset [92], the Cylinder Flow dataset [92], and the ScalarFlow dataset [34].

The Airfoil dataset is designed to simulate and predict the aerodynamics around the cross-section of an airfoil wing. It is based on compressible Navier-Stokes equations, modeling the evolution of momentum and density fields around an airfoil. This dataset is particularly relevant for studying flow separation, lift, and drag forces, which are crucial in aerospace engineering and aerodynamic optimization. The dataset was generated using the SU2 solver, a widely used open-source computational fluid dynamics (CFD) solver for compressible flows. The simulation operates on a 2D Eulerian triangular mesh, where each node encodes relevant flow field quantities such as momentum, density, and pressure. The mesh is highly irregular, allowing for high resolution near the airfoil surface while maintaining a coarser resolution in less critical regions.

The Cylinder Flow dataset is a benchmark dataset that models incompressible fluid flow around a cylinder. It is widely used in fluid dynamics studies to analyze vortex-shedding phenomena, such as the Kármán vortex street, which occurs when a fluid flows past a bluff body at moderate Reynolds numbers. Understanding these dynamics is essential for applications in structural engineering, heat transfer, and turbulence modeling. The dataset is generated using the COMSOL solver, which provides high-accuracy finite-element solutions for incompressible Navier-Stokes equations. The simulation is carried out on a 2D Eulerian triangular mesh with an irregular node distribution to capture fine-scale turbulence near the cylinder while maintaining computational efficiency in the outer flow regions.

The ScalarFlow dataset is a large-scale volumetric dataset that captures real-world scalar transport flows, primarily focusing on buoyancy-driven smoke plumes. Unlike many synthetic datasets generated through traditional computational fluid dynamics (CFD) simulations, ScalarFlow reconstructs real-world smoke flows from physical experiments, providing a unique opportunity to bridge the gap between simulated and real-world fluid phenomena. This dataset was created using a multi-view capture system and physics-based tomographic reconstruction. The experimental setup includes a fog machine inside an insulated container, controlled heating elements, and multiple cameras arranged around the scene. The cameras record video streams of the smoke plume from different angles, which are then processed through an optimization-based reconstruction framework to obtain 3D velocity and density fields. The ScalarFlow dataset is particularly useful for studying turbulent mixing, advection, and buoyant fluid behavior.

2.3.2.2 Evaluation Protocol. Evaluating machine learning-based CFD surrogate models requires an evaluation protocol to ensure that the models accurately approximate single-step predictions and maintain stability and consistency over multi-step simulations. To this end, we employ a rollout experiment to assess long-term prediction performance and normalized Root Mean Squared Error (nRMSE) as the primary quantitative metric for accuracy evaluation.

A key requirement for any CFD surrogate model is its ability to maintain accuracy over extended time horizons. To systematically evaluate this, we perform rollout experiments, where the model is initialized with a known flow state and iteratively predicts the next time step using its outputs as inputs for future predictions. This process simulates how the model would behave in real-world scenarios where ground-truth data is unavailable beyond the initial conditions. By comparing the predicted flow evolution against ground-truth CFD simulations over multiple rollout steps, we quantify the extent to which errors compound over time. This experiment provides critical insights into model stability, robustness, and generalization beyond short-term predictions.

We adopt the normalized Root Mean Squared Error (nRMSE) on all output channels to measure per-step prediction accuracy. This provides a scale-independent measure of prediction quality.

Given the true flow field values y and the predicted values $\hat{y}$, the nRMSE is computed as:

$$\text{nRMSE} = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}}{\max(\mathbf{y}) - \min(\mathbf{y})} \tag{2.1}$$

, where N represents the total number of nodes in the CFD mesh. Unlike standard RMSE, nRMSE normalizes errors by the dynamic range of the data, making it more interpretable across different flow conditions and datasets.

Combining rollout experiments with nRMSE evaluation ensures a rigorous assessment of our learned CFD surrogate models. Specifically, we analyze:

- Short-term accuracy: How well the model approximates the next-step predictions.

- Long-term stability: Whether errors accumulate over multiple prediction steps.

- Generalization across conditions: Whether the model remains robust under different initial conditions and flow regimes.

This evaluation protocol provides a holistic framework for benchmarking the effectiveness of machine learning-based CFD models. It ensures both accuracy and stability for real-world deployment.

2.3.3 Experimental Setup for Spatiotemporal Analysis Models

2.3.3.1 Datasets. For spatiotemporal analysis models, all experiments are conducted on three publicly available real-world datasets: the SmartAQnet dataset (SAQN) [63], the NOAA Aircraft-Based Observation dataset (ABO) [86], and the Copernicus In-situ Marine Observation dataset (Marine) [105].

The SAQN dataset is a typical example of the traditional fixed-location sensor network dataset. It is collected entirely from fixed-location sensors. The SAQN dataset uses SmartAQnet data from January 1, 2017, to December 31, 2021. The time interval of the Observation Frame is 1 hour. The observed area is a rectangular area within 14 kilometers north and east from 10.7992° E and 48.421° N. The target OP is PM10. Support OPs include PM2.5, temperature, relative humidity, air pressure, longitudinal wind speed, latitudinal wind speed, precipitation, and solar radiation.

The ABO dataset is exclusively composed of moveable sensor data, with observations recorded by airborne sensors installed on commercial aircraft. For the ABO dataset, we selected all observations in this dataset from July 1, 2001, to April 1, 2004, located in the range of 74.0° W to 75.0° W, and 40.0° N to 41.0° N. The time interval of the Observation Frame is 1 hour. The target variable is air temperature, and the support variables include wind speed and wind direction.

The Marine dataset combines observations from fixed-location (buoys) and moveable (ships) sensors. For the Marine dataset, we selected all observations in this dataset from January 1, 1900, to December 31, 2010, located in the range of 36.0° W to 11.0° W, and 31.0° N to 56.0° N. The time interval of the

Observation Frame is 4 hours. The target variable is water temperature, and the support variables include air temperature, air pressure, dew point, wind speed, and wind direction.

The following preprocessing steps are common to all the datasets:

- **Step 1**: Exclude outliers. In this step, we use the threshold method to exclude outliers that do not comply with physical laws and are too far distributed. The preprocessing code provides further detail.
- **Step 2**: Split the Observation Frames. We split the observations in the dataset into different Observation Frames according to the time intervals mentioned above.
- **Step 3**: Spatial aggregation. We further partition the horizontal space into a 250×250 grid for each Observation Frame. Then, we aggregate the readings with the same coordinates by averaging.
- **Step 4**: Filter the Observation Frames. We only retain Observation Frames that provide at least 5 nodes for training and one node for evaluation for all hold-out regions. For datasets that still have more than 20,000 graphs after filtering, we retain the 20,000 graphs with the latest timestamps.

2.3.3.2 Evaluation Protocol. Figure 2.3 illustrates our strategy for dividing the dataset.

In the temporal dimension, we divide all the Observation Frames into three parts according to their temporal order: 60%, 20%, and 20% each, which are used in the model's training, validation, and testing, respectively.

In the spatial dimension, we divided the study area into four equal parts by dividing the length and width of the horizontal area equally. We adopt the leave-one-area-out cross-validation method and, in turn, use four areas as hold-out areas. The training and validation of the model are performed only with the label within the three non-hold-out regions, while the model is tested only with the label in the hold-out region. With the above evaluation strategy, we ensure that the models are tested only on Observation Frames and locations that have never been seen during training and validation.

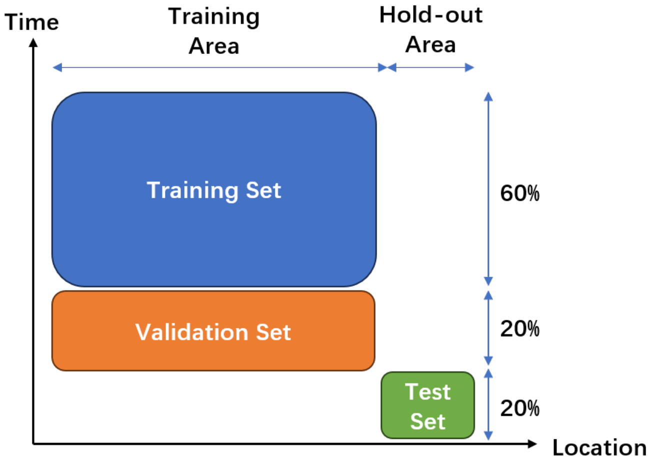


Figure 2.3: Our strategy for dividing the dataset. With this strategy, we ensure that the models are tested only on time frames and locations that have never been seen during training and validation.

Further, test locations might be densely surrounded by other sensors, making it hard to evaluate whether the model performs well in remote target locations. Therefore, when testing the model for each target location, in addition to testing with the complete Observation Frame, we also use two other Observation Frames that remove all nodes within 20 or 50 pixels by Manhattan distance of the target location, simulating the situation that target locations of different levels of remoteness.

As in other literature in the field, we choose the mean absolute error (MAE) and coefficient of determination (R^2) between the model output and the label (value of the corresponding target variable) as the metrics to evaluate the model performance. We first calculate the performance of four-fold leave-one-area-out cross-validations under each random seed and then calculate the mean and standard deviation of the results between different random seeds.

2.4 Chapter Summary

In this chapter, we formally define the problem to be solved, discuss the technical approach to solving the problem, and introduce the dataset and evaluation protocol used in the experiments.

3 Hybrid Sensor Network Datasets

This chapter first introduces a new urban air quality dataset (the SmartAQnet 2020 dataset) with a large span and high resolution in both time and space dimensions as a case study for the data product of the hybrid sensor network for urban aerosol monitoring. Then, we identify the key challenges in hybrid sensor network datasets.

3.1 Case Study: The SmartAQnet 2020 Dataset

3.1.1 Related Works

Air pollution causes severe damage to human health. The WHO Air Quality Guidelines [46] state that adverse health effects of air pollution can be observed not only in high exposures but also at very low concentration levels. Due to the large concentration of human activities in urban areas, information on urban air pollution is particularly interesting. However, fine-grained monitoring and forecasting of urban air pollution remain a major challenge.

Current urban air pollution models can be classified into two main types: physical and statistical models. The traditional process of the physical model is first to estimate the possible emission sources. Then take emission sources and meteorological data as inputs, input them into a series of physics equations, which simulate the transfer, diffusion, and chemical reactions of pollutants [55; 110]

Unlike physical models that pay more attention to physical and chemical rules, statistical models mainly focus on using methods such as machine learning to summarize the statistical characteristics of historical observation records [22; 108]. With the rapid development of machine learning technology, more and more researchers have paid attention to statistical models in recent years.

Training for good statistical models is becoming more and more data-hungry.

Traditionally, urban air quality data is usually collected by a few stationary, high-precision professional measuring stations [13]. They are accurate, well maintained, but expensive and need experienced personnel. Only relatively few organizations with sufficient technical and financial resources can establish such measurement networks. Although such stations can provide high-quality data, it is challenging to base fine-grained models on their data because of their scarcity in numbers.

However, in urban areas, factors related to air quality, such as human activities, meteorology, and land use, are highly complex and may change rapidly. In order to meet the current needs, the paradigm of air quality monitoring has started to shift towards monitoring urban air quality by deploying a large number of low-cost measuring sensors [12] to achieve higher spatial and temporal resolutions [66; 111]. Our measurement network, the Smart Air Quality Network (SmartAQnet), incorporates both data from high-quality measurement stations, as well as lower-cost and -fidelity measurement equipment.

3.1.2 The SmartAQnet 2020 Dataset

The Smart Air Quality Network (SmartAQnet) [15] concentrates on recording urban meteorology and aerosol measurement data in fine granularity using heterogeneous measurement technology. Since 2017, this project has collected over 300 million observations in the model region of the City of Augsburg, Germany, and this number is still rapidly increasing as time goes on.

The SmartAQnet 2020 dataset includes all the data collected by the SmartAQnet project during the time interval from January 1, 2017, to December 31, 2020. The dataset contains 248,572,003 observations collected in the model region of the City of Augsburg, Germany. These observations are recorded by over 180 individual measurement devices, including ceilometers, Radio Acoustic Sounding System (RASS), mid- and low-cost stationary measuring equipment using meteorological sensors and particle counters, and low-weight portable measuring equipment mounted on different platforms such as trolley, bike, and UAV. Various aerosol and meteorological features are measured and collectively referred to as observed properties. Table 3.1 below shows the number of observations included in each observed property. There are sometimes multiple observed properties in the Table corresponding to the same measured object because the SmartAQnet 2020 dataset comes from a large number of heterogeneous sensors.

Table 3.1: The number of observations included in each observed property

Abbreviation in Dataset	**Counts**	**Description**
saqn:op:absp	665,663	Attenuated Backscatter Profile
saqn:op:bc	27,048	Black Carbon
saqn:op:blh	1,262,184	Boundary Layer Height
saqn:op:ca	665,669	Cloud Amount
saqn:op:dp	4,470,887	Dew Point
saqn:op:globalrad	283,541	Global Radiation
saqn:op:hur	48,313,049	Relative Humidity
saqn:op:irbcc	30,696	Infrared Particulate Matter (IRPM)
saqn:op:mcpm	47,088	PM Total Mass Concentration
saqn:op:mcpm1	14,760,604	PM1 Mass Concentration
saqn:op:mcpm10	53,903,439	PM10 Mass Concentration
saqn:op:mcpm2p5	53,759,429	PM2.5 Mass Concentration
saqn:op:mcpm4	1,871,276	PM4 Mass Concentration
saqn:op:mcpmtotal	120,517	PM Total Mass Concentration
saqn:op:mcresp	120,519	PM4 Mass Concentration
saqn:op:ncpm	27,048	PM Total Particle Number Concentration
saqn:op:ncpm1	54,096	PM1 Particle Number Concentration
saqn:op:ncpm10	9,465,732	PM10 Particle Number Concentration
saqn:op:ncpm2p5	81,144	PM2.5 Particle Number Concentration
saqn:op:plev	5,556,495	Air Pressure
saqn:op:pnc0p2-1	177,075	Particle Number Concentration (0.02 - 1 µm)
saqn:op:precip	310,545	Precipitation
saqn:op:sigmaw	16,206	Sigma of the Vertical Wind
saqn:op:ta	48,526,008	Air Temperature
saqn:op:td	1,331,371	Temperature in Device
saqn:op:theta_a	16,229	Acoustic Potential Temperature
saqn:op:total	2,273	PM Total Particle Number Concentration
saqn:op:uvbcc	30,696	Ultraviolet Particulate Matter (UVPM)
saqn:op:wchill	15,104	Wind Chill
saqn:op:wdir	404,999	Wind Direction
saqn:op:wspeed	766,799	Wind Speed
saqn:op:zcb	665,663	Cloud Base Altitude
saqn:op:zcl	665,676	Cloud Layer Altitude

As mentioned above, the SmartAQnet project contains a large number of low-cost sensors. These sensors are cheap and simple for massive deployment, but on the other hand, their working conditions are not as stable as those expensive sensors. Therefore, the actual deployment scale of our project and the number of observations fluctuate over time. Figure 3.1 below shows how the number of observations included in each month in the dataset changes. Figure 3.2 shows the average number of daily available devices for each month.

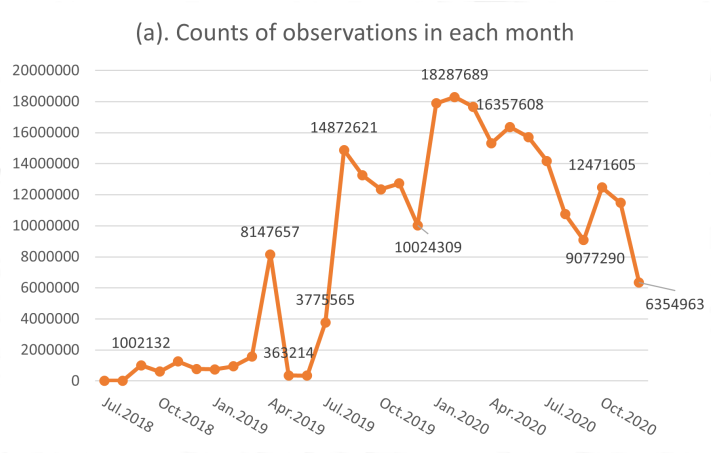


Figure 3.1: Counts of observations in each month.

So far, the measuring equipment involved in the SmartAQnet project could be categorized into four parts:

- High accurate scientific measurement technology
- Consumer-grade and Low-cost measurement sensors
- Location- and period-fixed Mobile measurements
- Intensive Sensing Campaigns

Each of these four parts has its own characteristics of time resolution, spatial resolution, and accuracy. As shown in Figure 3.3, they cooperated in providing a detailed and comprehensive observation system.

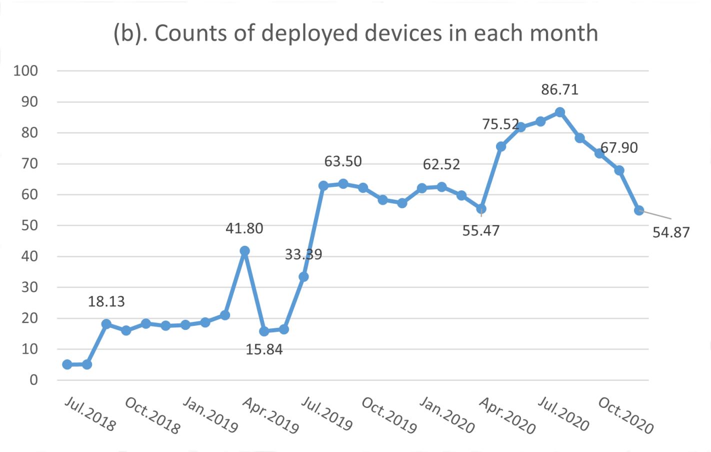


Figure 3.2: Average counts of deployed devices in each month.

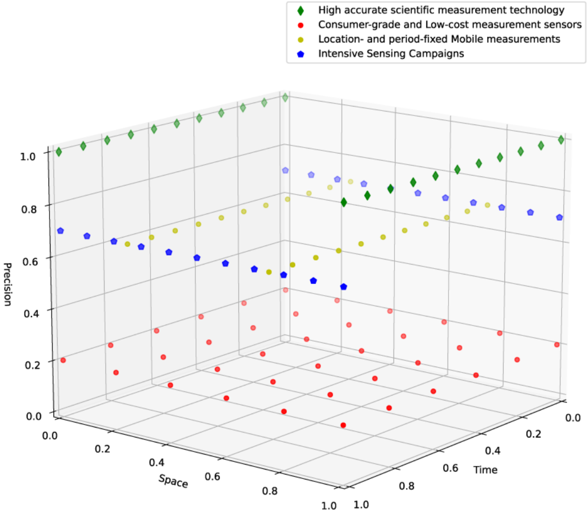


Figure 3.3: Characteristics of different parts in terms of time resolution, spatial resolution, and accuracy.

3.1.2.1 High accurate scientific measurement technology. SmartAQnet is integrated into the existing measurement network in the Augsburg area. In other words, it incorporates the publicly available data of high-precision measurement equipment provided by local authorities.

First of all, there are four state air quality monitoring stations provided by the LfU Bayern (Bavarian State Office for the Environment), which (among other parameters) collects PM10 readings.

The second existing measurement network is a ground-based remote sensing network consisting of a VAISALA CL51 ceilometer and a METEK Radio-Acoustic Sounding System (RASS) (see [36; 37]), both located on the campus of the University of Augsburg. This measurement network serves to observe cloud, wind, and temperature data by height profiles.

The third existing network is a local meteorological station network consisting of 7 stations, collecting various meteorology properties, such as Temperature, Wind, Pressure, Precipitation, etc.

The three aforementioned networks provide observations with very high accuracy, as well as satisfactory temporal resolution. However, the spatial resolution is relatively poor due to the sparse number of stations.

3.1.2.2 Consumer-grade and low-cost measurement sensors. The inexpensive sensors involved in the SmartAQnet project could also be briefly classified into two different precision levels: Consumer-grade sensors and low-cost sensors. Consumer-grade sensors can be seen as a compromise between high-precision measuring stations and low-cost sensors. In terms of price and maintenance costs, they are somewhere between the above two. At the same time, the precision of the data they can provide is also high enough to be used as a reference device. We deployed 6 Grimm EDM-164OPC Sensors during the project as consumer-grade sensors, which serve as reference devices of the Grimm sensor network.

Compared with high-precision stations and consumer-grade sensors, low-cost sensors reduce precision for lower price and maintenance costs, enabling them to be massively deployed to provide much higher temporal and spatial resolutions. In the SmartAQnet project, part of the low-cost sensors come from integrating existing local equipment, and a much more considerable amount was directly deployed during the project. We deployed 22 Grimm EDM-80NEPH Sensors and 35 Grimm EDM-80OPC Sensors. Together with the above-mentioned 6 Grimm EDM-164OPC Sensors, these sensors form the so-called Grimm network, in which low-cost sensors could make intelligent signal evaluation through extensive comparison measurements to reference devices. We also have 84 Crowdsensing Nodes composed of a Nova SDS011 Ultra-fine Particulate Sensor and a Bosch BME280 Sensor. All the sensors mentioned above can provide PM and basic meteorological observations, and they together constitute the main body of this part. In addition to these newly deployed devices, two existing devices provided by Helmholtz Zentrum München (HMGU) are integrated into the project, known as HMGU EPI PM Container and HMGU EPI Meteo Container.

In the SmartAQnet project, the locations of the deployed sensors were also carefully designed. We selected a rectangle area of about 4 × 6 km in the centre of Augsburg as the Central Activity Zone (CAZ), which northwest located at 48.39°N, 10.87°E and southeast located at 48.33°N, 10.92°E. We have consciously increased the deployment density of sensors in the CAZ since it contains three of the four high-precision stations so that the sensors deployed in this area can get better references. The following Figure 3.4 shows the locations of the CAZ and different kinds of stationary sensors.

3.1.2.3 Location- and period-fixed mobile measurements. Mobile measurements were carried out between December 2019 and September 2020 in a fixed period and location using two UAV systems. One is a self-constructed fixed-wing aerial vehicle, and the other is a rotorcraft of type DJI Matrice 600 pro. Both UAVs are equipped with a measuring device that integrates low-weight weather parameter sensors (Sensirion SHT75/SHT85) and particle counters (Alphasense OPC-N2/OPC-N3), special sensor inlets and pumps are installed to reduce the influence of the UAV. Thus, they could detect relative humidity, temperature, and PM concentrations data.

The mobile measurement data provided by the UAV observes the detailed three-dimensional dynamics of the lower atmosphere with fine granularity,

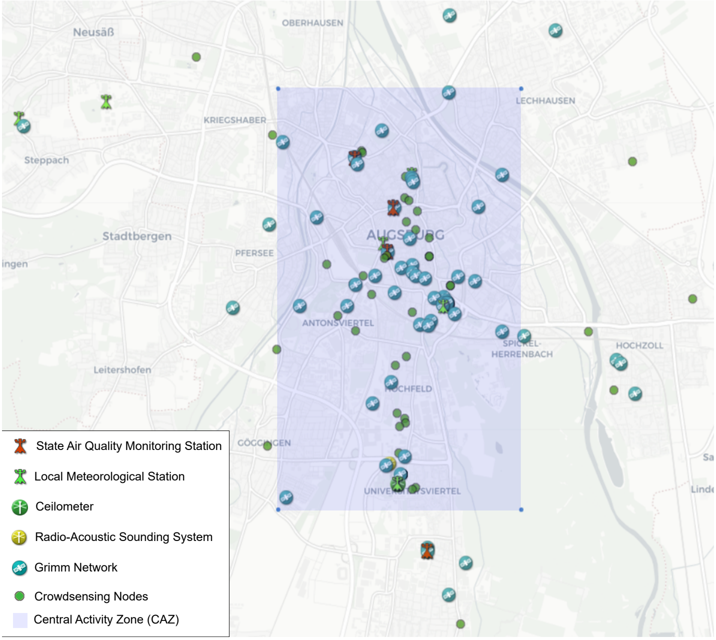


Figure 3.4: Locations of the CAZ and different stationary sensors in and around the City of Augsburg, Germany.

which can be used as a powerful supplement to the remote sensing measurement provided by the ceilometer and RASS. However, due to the limitation of UAV load capacity, these measurement data can only be collected with low-weight sensors, so the observation precision cannot match with the highly accurate scientific measurement technology. In addition, since the UAV flight requires the operation of the pilot on the ground, it can only be carried out regularly at a certain frequency, thus cannot fully cover the time dimension.

3.1.2.4 Intensive Sensing Campaigns. During the data collection period of SmartAQnet, several intensive sensing campaigns were held. Mobile measuring devices were installed on trolleys and bicycles during the campaigns and were carried through the city by participating personnel.

Between August 2018 and June 2020, several intensive sensing campaigns were carried out using trolleys equipped with portable sensors. The trolleys are equipped with GPS (GPSMAP 64s, Garmin, USA), which records the position with the 1-second resolution, and is equipped with a variety of portable PM sensors, such as DustTrak DRX Aerosolmonitor, P-Trak Ultrafine Particle Counter 8525, Grimm 11e, Aethlabs microAeth MA200, Hand-Held Condensation Particle Counter Model 3007, Testo DISCmini, Aerocet 531S, etc. Figure 3.5(a) below shows the route of the trolleys' measurements.

Since January 2020, we have introduced another mobile measurement method. Backpacks equipped with GPS, low-weight weather parameter sensors (Sensirion SHT75/SHT85), and particle counters (Alphasense OPC-N2/OPC-N3) were installed on bicycles for mobile measurement. Measurements made with bikes are more frequent compared to measurements performed with trolleys. One could find multiple measurement activities in most months of 2020. Figure 3.5(b) shows an example route from the bike measurements.

The coverage in the time dimension is more limited due to the effort required to organize activities. However, this part of the data has extremely high coverage in the space dimension.

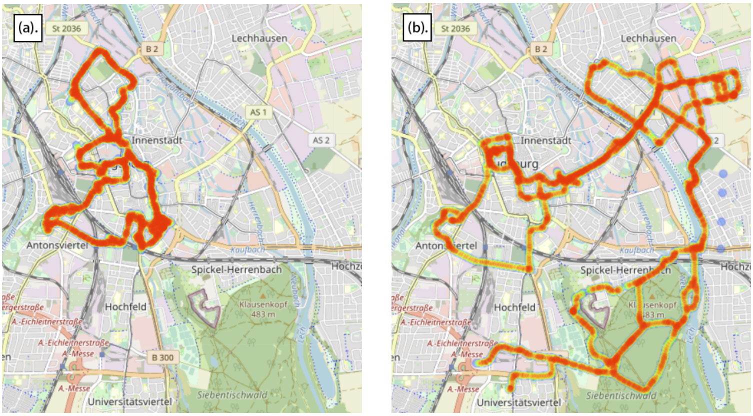


Figure 3.5: (a). Route of the trolley measurements. (b). An example route from the bike measurements.

3.1.3 Opportunities

In this section, we would like to discuss the further opportunities we believe the SmartAQnet 2020 dataset provides. Work already done based on data from the SmartAQnet project includes the performance evaluation of low-cost PM sensors [17], the development of novel calibration approaches [103], spatial modeling [106] and interpolation [120] approaches, and higher-level applications such as air-quality-based bike routing [50]. Even meta-level discussions have been informed by the experiences in collecting distributed air quality data from heterogeneous sensors, such as work towards sustainable business models for high-resolution air quality assessment [102].

The first opportunity the SmartAQnet 2020 dataset provides is modeling urban air quality. In our dataset, we have a large number of observations, observing the model area with high resolution. It can meet the requirements of statistical modeling very well. On the other hand, since we have also recorded various meteorological data, it could also be used for physical modeling.

Secondly, the dataset can also be used to evaluate Spatial-temporal interpolation algorithms. The data provided by the high-precision measuring station and the intensive sensing campaigns offer multiple possibilities for evaluation.

Thirdly, the dataset provides good opportunities for understanding the performance of low-cost sensors in the field. SmartAQnet features long-term, high-volume, low-cost sensor usage. Therefore, the dataset holds the possibility to illustrate problems or limitations in data quality [16] when deploying low-cost sensors. It can potentially also be used to benchmark the real-world applicability of different existing [14] or newly proposed data cleaning or distributed calibration methods for low-cost sensors (e.g. [28; 44; 78].

Fourth, the dataset could be used for understanding how the Coronavirus (SARS-CoV-2) – or rather the accompanying changes in urban activity – affected urban air quality. Our dataset covers both the first wave of the Corona period, which means the second half of 2020, and the non-Corona-period. During the Corona period, with different restrictive policies in effect, reduced human activity possibly affected the distribution of particulate matter. By analysing the difference between these two periods, we could gain a deeper understanding of such effects.

3.1.4 Accessing the Dataset

The SmartAQnet 2020 dataset is publicly available with a CC BY 4.0 Attribution license. The latest version of the dataset as of this writing can be accessed with the following DOI: 10.35097/540. All works that make use of the SmartAQnet 2020 data should include a reference using that DOI, as well as giving scholarly credit by citing both the SmartAQnet project (DOI: 10.1117/12.2282698) and this dataset paper itself (DOI: 10.14644/dust2021.001).

Since most of the sensors used in the SmartAQnet project are still in operation, we will release our new datasets at a certain frequency as a supplement to the SmartAQnet 2020 Dataset. Information about the latest release of the dataset can be obtained by visiting the official homepage of the SmartAQnet project (www.smartaq.net).

The SmartAQnet 2020 dataset itself only includes PM- and meteorology-related records. Other datasets of factors closely related to the distribution of urban air pollutants, such as land use, traffic volume, etc., are not released together for distinct reasons. If there is a need for these data, we are willing to assist within our capacity.

3.1.5 Summary

In this section, we presented the SmartAQnet 2020 Dataset. It was collected in the model region of Augsburg, Germany between 2017 and 2020 and contains 248,572,003 observations recorded by over 180 individual devices, including ceilometers, Radio Acoustic Sounding System (RASS), mid- and low-cost stationary measuring equipment using meteorological sensors and particle counters, and low-weight portable measuring equipment mounted on different platforms such as trolley, bike, and UAV.

We have used the dataset ourselves for a variety of analyses, including the development and evaluation of modeling, spatial interpolation, and distributed calibration approaches. We provide the dataset to the public under a permissive open data license as an opportunity to develop or benchmark existing or novel approaches based on heterogeneous, real-world air quality data.

3.2 Key Challenges in Hybrid Sensor Network Datasets

Based on our experience analyzing multiple hybrid WSN datasets, we summarise the challenges of hybrid WSN datasets as follows:

3.2.1 Heterogeneity

Hybrid WSN datasets are heterogeneous in several ways, typically including but not limited to the following:

- **Heterogeneity of sensor models**: A hybrid WSN may be a mixed network of various sensor models. This makes it difficult to obtain readings of all observed properties at the same coordinate simultaneously, which is often one of the underlying assumptions of studies based on traditional sensor networks.

- **Heterogeneity of sensor network structure**: Unlike long-lived traditional measuring stations, low-cost sensors are unstable. Sometimes, deploying new ultra-low-cost sensors is even more affordable than retrieving and repairing old ones. These factors keep the spatial structure of hybrid WSNs changing. Figure 3.6(a) illustrates how the total device amount of one of such sensor networks changes over time, while Figure 3.6(b) shows when part of sensors in the network successfully returned data and did not. It is easy to figure out that the architecture of hybrid WSNs is precarious. Furthermore, in addition to stationary sensors, some sensor networks also partially [63; 105] or fully [66] employ mobile sensors, which further enhances the heterogeneity of the spatial structure of the sensor network.

- **Heterogeneity due to low-power wireless communication protocols**: Deployment of traditional sensors often faces administrative difficulties, such as applying for land, power supply, and network access from local administrations. As a result, many WSNs turn to using low-power wireless communication technologies such as LoRaWAN, Zigbee, BLE, Z-Wave, etc. These communication protocols allow sensors to operate on batteries alone for a considerable period and send their observations to the data center wirelessly. The cost of this is usually a restricted uplink bandwidth and transmission time window. Even if the network delivers excellent completeness data at the end, the real-time heterogeneity induced by these protocols must be considered when considering the actual deployment of the model for real-time usage.

3.2.2 Uncertainty and error

The uncertainty of hybrid WSN datasets is reflected in the following aspects:

- **Uncertainty and error in sensor readings**: The accuracy of a sensor, not only in terms of its accuracy on its observed properties but also its position recorded by the mounted GPS module, is generally related to its price level. Some studies have also pointed out that how low-cost sensors are assembled and the environment they operate can also harm their measurements. In severe cases, it can even return only qualitative results [17].

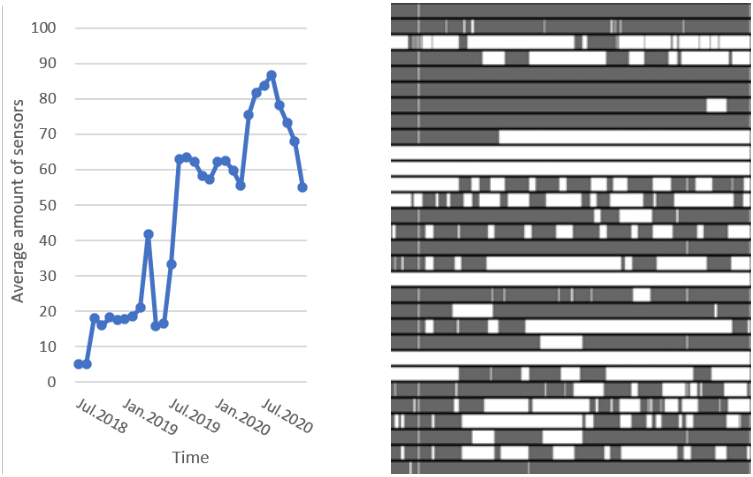


Figure 3.6: (a). The curve of the monthly average active sensors in the network in the SmartAQnet dataset [63] (Jul. 2018 to Dec. 2020). (b). Daily activity status of a subset of low-cost sensors in the SmartAQnet dataset (2021.01.01 to 2022.01.01). Each row represents a sensor, with white indicating no readings collected in the day and black indicating the opposite.

- **Uncertainty and error in manual logging**: Maintaining a WSN is a complex, long-term task that requires much manual logging during the installation, repair, and transfer of sensors. For example, most ultra-low-cost sensors are not equipped with GPS modules, and their deployment locations in the network rely entirely on manual records. Considering the operation period of WSNs is often measured in years, human errors are almost unavoidable, and a significant portion is difficult to identify and fix in the quality check.

- **Uncertainty from unobserved influencing factors**: Another primary source of uncertainty arises from unobserved external factors that influence aerosol distribution but are not explicitly recorded in sensor network datasets. Urban aerosol levels are affected by a highly dynamic and complex set of variables, including local meteorological conditions, transient pollution sources, and microscale urban structures. However, due to limitations in sensor placement, cost constraints, and the inherent sparsity of observation networks, many of these factors remain unmeasured or only partially captured, introducing hidden sources of error into predictive models. For instance, sudden changes in wind direction or turbulence effects near buildings can cause sharp variations in aerosol concentrations that sensors fail to capture adequately. Similarly, temporary pollution sources, such as construction activities, industrial emissions, or traffic congestion, may contribute significant short-term fluctuations that are not reflected in standard monitoring data. Even in well-instrumented networks, localized effects like street-canyon turbulence or vegetation filtering introduce variability that is difficult to model without additional environmental context. Such unobserved influencing factors make aerosol prediction inherently ill-posed, as models must infer missing information based solely on available, potentially biased observations. This challenge underscores the need for uncertainty-aware modeling approaches, such as Bayesian deep learning, uncertainty quantification frameworks, and hybrid physics-informed models, to ensure robust predictions despite incomplete observational data. Without explicitly accounting for these hidden variables, models may exhibit systematic biases or erroneous attributions, reducing generalization performance in real-world urban environments.

3.2.3 Fusion of support observed properties

In addition to the primary observed property, hybrid WSN datasets also observe many other support observed properties that may relate to it. Fusing the knowledge these support observed properties provide will help model the primary observed property. However, it is also important to note that in some rapidly changing and complex observed systems (e.g., in urban environments), we still need more prior knowledge of how these support observed properties affect our tasks.

3.2.4 Generalization capability

WSNs can only provide observations near the sensor location, which usually only represents a tiny fraction of the entire observation area. If we further require a reasonably reliable accuracy level for model validation, even fewer locations are available. This also places demands on the generalization capabilities of the model. The model should either overcome the bias introduced by the limited validation locations or could be validated with noise-containing data.

3.3 Chapter Summary

In this section, we use a real-world hybrid sensor network dataset (SmartAQnet 2020) as a case study. By introducing its situation, we give readers a basic understanding of the current status of urban aerosol monitoring technology. Subsequently, we summarize the challenges posed by hybrid sensor network datasets to spatiotemporal analysis models, thereby pointing out the direction for developing urban aerosol distribution prediction systems.

4 Data Augmentation

Data augmentation plays a crucial role in mitigating the limitations of real-world sensor network datasets, particularly in hybrid sensor networks where data is often sparse, noisy, and unevenly distributed. Unlike domains such as computer vision or natural language processing, where augmentation techniques like flipping, scaling, or synonym replacement can be applied with minimal risk, urban aerosol distribution modeling presents unique challenges. The physical nature of geospatial data makes it highly sensitive to absolute positioning, orientation, and environmental dependencies, rendering traditional augmentation strategies ineffective or even detrimental.

Moreover, the complexity of urban atmospheric dynamics introduces additional constraints on artificial data generation. Aerosol dispersion is governed by fluid mechanics, meteorological conditions, and emission sources, making it difficult to synthesize physically realistic data without introducing significant biases. To address these challenges, this chapter explores advanced data augmentation strategies specifically designed for urban aerosol prediction.

4.1 Development of a numerical CFD model for pollutant dispersion at an urban traffic hotspot

Please note that Section 4.1 on CFD modeling and the corresponding paper were primarily contributed by my project colleague, Dr. Giannis Ioannides. My primary responsibility was the collection and organization of the data. We present the CFD modeling process and key results to provide an overview of the methodology and current research status of using CFD to generate synthetic urban aerosol data. We sincerely acknowledge Dr. Giannis Ioannides' contributions.

Accurately representing urban air pollution for multiple contaminants is essential to understand pollution levels comprehensively. However, previous studies have not employed CFD simulations to model the dispersion of CO, NOx, and particulate matter (PM) directly from vehicular sources, likely due to the scarcity of emission data. This work simulates pollutant dispersion in a high-traffic urban area, incorporating hourly emission rates and wind conditions to achieve high temporal resolution. Unlike methods relying on daily averages, this approach captures hourly variations, enabling the identification of peak and low pollution periods throughout the day. The validated CFD model constructs a detailed three-dimensional concentration gradient of pollutants, providing spatially and temporally refined insights for practical applications.

4.1.1 Related Works

Understanding airflow dynamics is essential for accurately modeling pollutant dispersion in urban environments. Many studies have focused on turbulent flow and gas dispersion within idealized street canyons, primarily investigating how urban geometry influences pollutant transport. For instance, Yazid et al. [138] found that buildings with sharp roofs enhance pollutant recirculation, while Tee et al. [116] highlighted the impact of the turbulent Schmidt number (Sct) on gas concentration predictions. Montazeri et al. [83] demonstrated that balconies can significantly alter wind pressure distributions, and Su et al. [113] reported up to a 27% increase in pollutant concentrations due to tree planting in streets. While these studies provide valuable insights, they rely on simplified environments without incorporating real meteorological or emission data, limiting their applicability for assessing actual urban pollution exposure.

Applying CFD to real-world urban pollutant dispersion requires detailed data on street and building geometry, emission sources, and meteorological conditions. Although some studies have attempted to simulate traffic-related pollution in realistic settings, they have notable limitations. Lauriks et al. [61] analyzed NO2 and PM10 dispersion in Antwerp but only considered a single wind direction. Sanchez et al. [100] used a RANS-based approach to study NOx in Madrid under different traffic scenarios but excluded other pollutants. Rivas et al. [96] modeled NO and NO2 dispersion across Pamplona, achieving a moderate correlation (0.68) with observed hourly values but without considering additional pollutants. Akhatova et al. [1] examined CO dispersion in Astana using seasonal averages, lacking the temporal resolution needed to capture daily variations. These studies are constrained by two key factors: their focus on a limited set of pollutants and the absence of high-resolution, full-day temporal analyses, preventing a comprehensive depiction of urban air pollution.

4.1.2 Methodology

4.1.2.1 Area of interest and geometrical model. The simulation area covers a $650 \times 1050\ m^2$ region on the northern edge of Augsburg's city center. It includes buildings averaging 18 m, with the tallest structure, a chapel, reaching 56 m. The primary traffic corridor extends from Prinzregentenstraße (Point A) to Pilgerhausstraße (Point B). An air quality monitoring station, operated by the Bavarian State Office for the Environment (LfU), is located at (10.896, 48.370) along Karlstraße road, 2.5 m above ground level. This station records hourly concentrations of NO, NO_2, CO, $PM10$, and O_3 in $\mu g/m^3$.

The 3D city model was sourced from OpenStreetMap (OSM). Traffic-related pollutant dispersion is simulated by placing emission sources along the main road, covering the air quality station's vicinity. These sources, modeled as rectangular sections with an 8 m width (matching the road width), represent vehicle-emitting regions. The digital model (Figure 4.1b) includes seven emission sources distributed from Point A to Point B.

To define the computational domain, Blocken et al. [8] recommend a minimum height of 6H and an upstream distance of 5H, where H is the tallest building's height. This study employs a 350 m domain height (>6H) and 300 m clearance from boundaries to the nearest buildings (>5H), resulting in a total domain size of $1250 \times 1650\ m^2$. Figure 4.1c illustrates the domain's dimensions and boundary orientations (N, E, S, W).

4.1.2.2 Traffic emissions. Traffic emission rates for CO, PM, and NOx serve as inputs for the model. Emissions are calculated based on Average Daily Traffic Volumes (ADTV) for different road segments in Augsburg, georeferenced by road IDs, which provide vehicle count data for September 2018. The

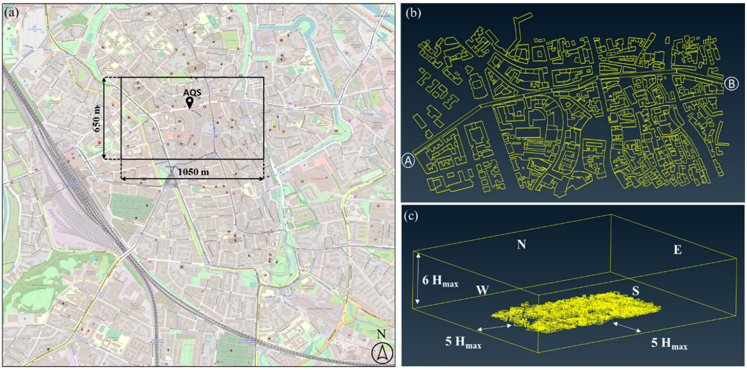


Figure 4.1: Surrounding environment containing the entire Augsburg city center (a), closeup to a case study area with emission sources (b) and computational domain dimensions (c)

COPERT Street software estimates emissions, considering the German vehicle fleet for 2018, with an assumed average speed of 30 km/h. The seven emission sources defined in Section 4.1.2.1 correspond to ten distinct road IDs.

4.1.2.3 Meteorological conditions. Meteorological data is a crucial input for air quality modeling. Wind speed and direction during the simulation period are obtained from a sensor located southeast of Augsburg's city center, part of the smartAQnet network. SmartAQnet is a collaborative sensor network in Augsburg, combining high-precision government measurement stations with many mid- and low-cost sensors operated by citizens and researchers. The network monitors 30 meteorological and aerosol parameters, including temperature, air pressure, humidity, precipitation, wind speed, and wind direction. It spans a 16×16 km^2 area, with a denser 6×4 km^2 section covering most of the city center.

4.1.2.4 CFD model setup.

Numerical model. The pollutant dispersion model is implemented using the open-source CFD software OpenFOAM. The built-in numerical solver simpleFoam employs the RANS equations to compute the velocity field [90]. To account for passive pollutant dispersion, the passive scalar transport equation (Equation 4.1) is integrated into a modified version of simpleFoam, which is then compiled into the system as a customized solver [35; 82].

$$\frac{\partial C}{\partial t} + \frac{\partial(\bar{u_J}C)}{\partial x_j} - \frac{\partial}{\partial x_j}\left((D_t + D_m)\frac{\partial C}{\partial x_j}\right) = 0 \tag{4.1}$$

D_m denotes molecular diffusion. For CO, the molecular diffusion coefficient at 20°C is 2.08×10^{-5} m^2/s [25]. NOx emissions, primarily NO, undergo partial oxidation to NO_2 in the atmosphere [96]. This study modeled the NOx mixture as NO_2 with a diffusion coefficient of 1.56×10^{-5} m^2/s. PM dispersion is modeled using a diffusion coefficient approach.

D_t represents the turbulent diffusion coefficient, calculated from Equation 4.2 by setting the turbulent Schmidt number (Sc_t) to 0.7 [119]. The kinematic viscosity of air under atmospheric conditions is predefined as 1.5×10 m^2/s at the domain boundaries. The turbulent viscosity v_t is updated at each timestep to characterize airflow within the simulation domain.

$$Sc_t = \frac{v_t}{D_t} \tag{4.2}$$

The standard k-ε turbulence model is used for the RANS simulations. As for the atmospheric velocity profile, Equation 4.3 refers to the profile set on the inlets of each case. u* refers to the friction velocity, κ to the Von Karman constant valued at 0.41, and z_0 is the aerodynamic roughness length at 2 m for urban environments [95].

$$U(z) = \frac{u^*}{\kappa} \ln\left(\frac{z + z_0}{z}\right) \tag{4.3}$$

Computational mesh. The CFD model utilizes a tetrahedral unstructured grid with a growth rate of 1.2 and a maximum skewness value of 0.9. To assess convergence sensitivity, two grids were developed: a medium mesh with 9 million cells and a fine mesh with 28 million cells. The medium mesh has resolutions of 2 m for buildings, 0.5 m for emission sources, and 15 m at domain boundaries, while the fine mesh refines these to 1 m, 0.25 m, and 15 m, respectively. Simulations were conducted with both grids to evaluate convergence. No significant differences in residuals were observed, indicating that neither grid had a clear advantage in solution accuracy. All simulations were executed on a computing node with Intel(R) Core (TM) i9-10980XE @ 3.00GHz CPUs using the same number of processors. Computation times were 651 minutes for the medium mesh and 2165 minutes for the fine mesh (Table 1). Due to its significantly lower computational cost—solving cases about three times faster—the medium mesh was chosen for all subsequent simulations.

4.1.3 Results

To validate the model, simulated pollutant concentrations are compared hourly with measurements from an air quality station along the main road. To isolate urban emissions, background concentrations from a station 5 km south of the study area are subtracted [51]. Since exhaust PM is primarily fine, a direct comparison with PM10 is not feasible. Instead, traffic-related PM contributions are estimated using a factor of 8% [93], while NOx and CO contributions are scaled using 54% and 88.5%, respectively [109].

Figure 4.2a shows that simulated NOx closely follows observed trends, with an average daily deviation of 14.3%. Both exhibit low concentrations in the morning, peaks at noon and afternoon, and a decline after 22:00, typical of weekend traffic patterns. For CO (Figure 4.2b), the model captures trends well, though with a daily deviation of 43%. CO sensor readings are quantized at 100 $\mu g/m^3$, while the CFD model provides higher precision.

Figure 4.3 shows a PM peak between 03:00–10:00 unrelated to traffic emissions, likely due to meteorological effects such as high humidity and low wind speeds [101]. Given these uncertainties, this period is excluded from PM analysis. After 10:00, simulated PM deviates by 18.57% on average but follows observed trends, with peaks in the afternoon.

Overall, the CFD model successfully captures traffic-driven pollution patterns, producing results consistent with NOx, CO, and PM measurements, particularly in periods where traffic dominates.

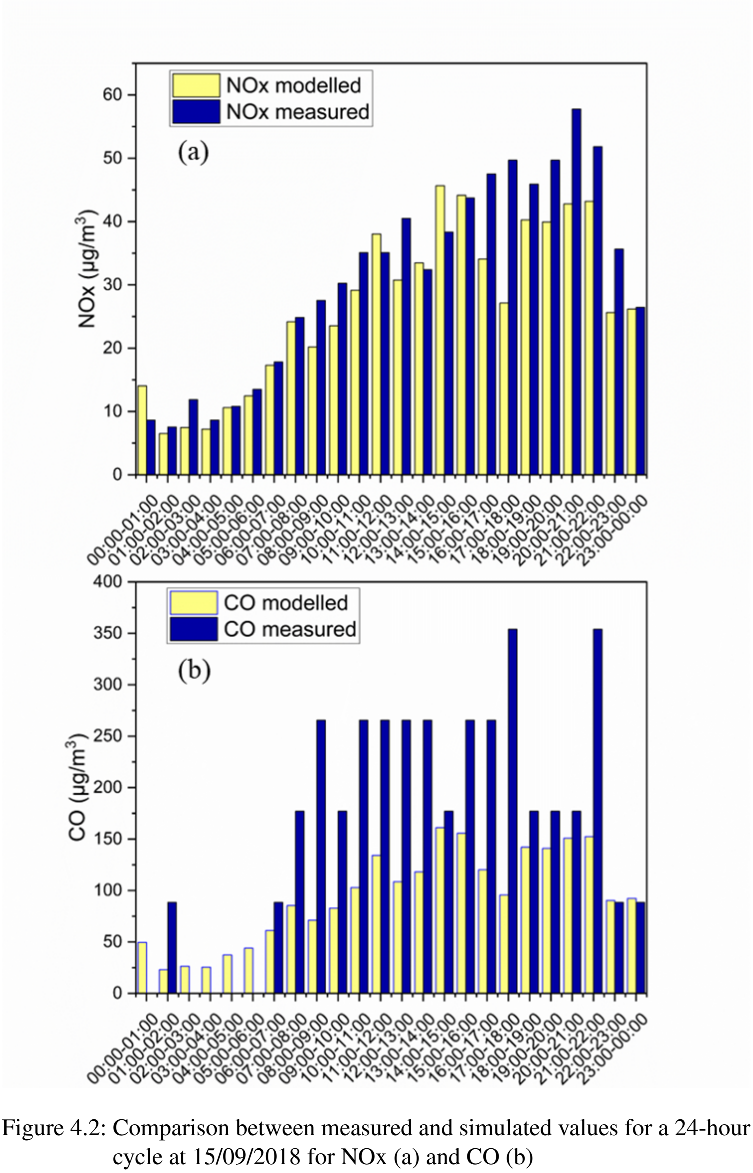


Figure 4.2: Comparison between measured and simulated values for a 24-hour cycle at 15/09/2018 for NOx (a) and CO (b)

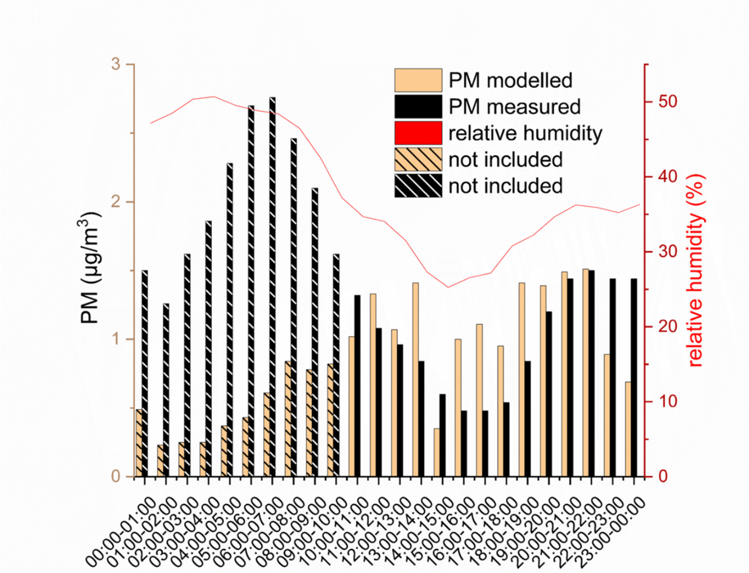


Figure 4.3: Comparison between measured and simulated Particulate Matter concentrations (left Y-Axis). Relative humidity on the corresponding period (right Y-Axis)

4.1.4 Summary

Urban air quality monitoring is increasingly critical, and modern tools enable creating high-precision digital networks. This study developed a detailed 3D model to map CO, PM, and NOx concentrations in a high-traffic urban area using CFD simulations over 24 hours. The open-source CFD code OpenFOAM was used to model pollutant dispersion from vehicular activity. Model validation against high-precision AQ station data showed low deviations from measurements, with statistical analysis confirming acceptable performance. The results closely follow traffic-driven urban pollution trends, demonstrating that the developed solver effectively simulates gaseous and particulate pollutant dispersion with reliable accuracy.

4.2 FlowCluster: Enhanced GNN-Based CFD Surrogate Model via Adaptive Graph Clustering

Computational Fluid Dynamics (CFD) simulations are widely used across various fields, including aerospace, automotive engineering, and environmental science [124]. Traditional CFD solvers typically rely on numerical methods such as the Finite Volume Method (FVM) [4] and the Finite Element Method (FEM) [149], which, despite providing high-accuracy fluid simulations, are extremely computationally expensive. As a result, accelerating CFD computations while maintaining accuracy has become a significant research focus.

In recent years, data-driven CFD surrogate models have emerged as a promising alternative. These models leverage deep learning techniques to capture the spatiotemporal evolution of CFD variables in a data-driven manner, significantly reducing computational costs during inference. In particular, since CFD data is typically represented as physical field variables on irregular meshes (Figure 4.7, top), graph neural networks (GNNs) have gained significant attention in CFD surrogate modeling [9; 92; 125]. Unlike traditional convolutional neural networks (CNNs), GNNs can directly utilize the graph structures defined by irregular meshes to propagate and aggregate information, making them a compelling approach for efficient CFD modeling.

Figure 4.5 illustrates the working mechanism of message-passing GNNs.

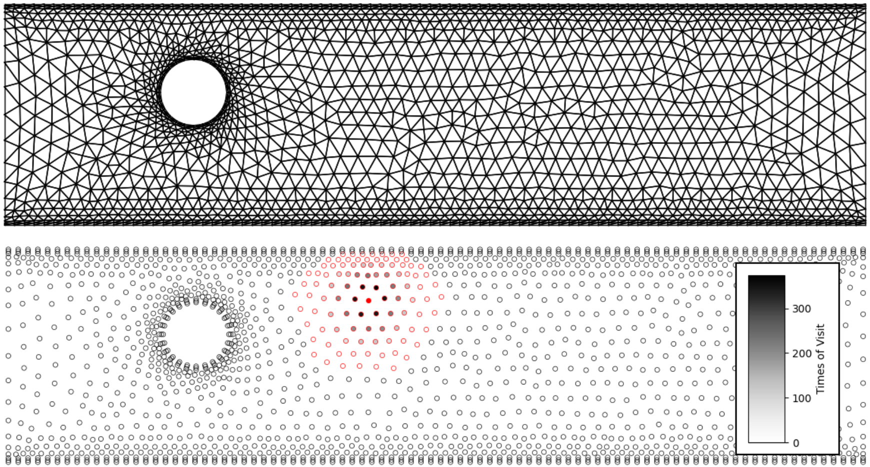


Figure 4.4: (Top) An example of the unstructured mesh from the "cylinder flow" dataset. (Bottom) For a specific central node (highlighted in red), the range of accessible nodes and their visit frequency during five layers of GNN message passing within the above mesh is illustrated. Nodes that were visited at least once are outlined in red, and the fill color of these nodes indicates the number of visits. In this scenario, the most frequently visited node was accessed 374 times by the central node.

Each message-passing layer aggregates information from a node's one-hop neighbors and updates the central node's representation. By stacking multiple message-passing layers, the central node can gradually access information from multi-hop neighbors, which is crucial for learning long-range dependencies. However, this makes GNN performance highly dependent on the underlying graph structure. On the one hand, increasing the number of message-passing layers expands the receptive field, allowing nodes to capture information from distant neighbors. On the other hand, as more layers are stacked, information from distant nodes undergoes excessive aggregation, making it indistinguishable by the time it reaches the central node. This phenomenon, known as over-smoothing, is particularly severe in CFD applications due to the high resolution of CFD meshes, which are designed to ensure numerical accuracy and stability.

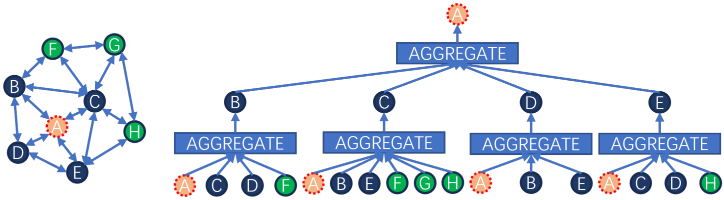


Figure 4.5: GNNs need several layers of message passing to gather information from distant nodes. However, graph structures based on triangle meshes result in significant over-smoothing.

Figure 4.7 (bottom) illustrates the severity of this issue. When five message-passing layers are stacked, the receptive field of a central node (marked in red) expands to include all nodes outlined in red. The color of each outlined node represents the number of times its information has been accessed during message passing. While the peripheral nodes are accessed only a handful of times, specific nodes within the receptive field are visited as many as 374 times, highlighting the inefficiency and imbalance in information propagation.

Clearly, an optimized graph structure is needed to improve the efficiency of information propagation. Several existing studies have explored techniques such as attention mechanisms [11; 49], node clustering [48; 49], boundary encoding [125], or U-Net architectures [19] to enhance message passing in GNN-based CFD surrogate models. However, these methods primarily rely on static graph optimization, where a one-to-one mapping between the mesh and the optimized graph structure is pre-computed. The same graph structure is used throughout training and inference, regardless of how the physical field evolves.

The limitation of static graph optimization is that it lacks adaptability to different flow states. For example, when using mesh-based static graph clustering, the parts of a specific flow structure (such as a vortex) may be broken down and assigned to different hyper-nodes. This results in the encoder and decoder components of the hyper-nodes encountering overly complex and variable input patterns, ultimately degrading the model's performance. However, if the current flow field state is considered and these parts are clustered into the same hyper-node, the patterns for the encoder and decoder components will become more stable. In this work, we propose an adaptive framework where the optimized graph structure is determined not only by the mesh itself but also by the current state of the physical field. This enables the message-passing mechanism to adjust dynamically according to flow variations.

Our work makes the following key contributions:

- **Adaptive graph node clustering**: We propose FlowCluster, a GNN-based CFD surrogate model. The main workflow of FlowCluster is inspired by the EAGLE model [49], with several key improvements based on analysing its performance in experiments. The most significant enhancement is the introduction of adaptive graph node clustering based on the current flow field state. Unlike the EAGLE model, which relies on a precomputed, fixed node clustering based solely on the mesh, our approach dynamically computes node clusters during training and inference phases by incorporating real-time flow field information.

- **Efficient clustering based on anchor nodes**: One of the main reasons that existing methods rely on precomputed static optimization is the high computational cost of graph structure optimization. Since adaptive graph node clustering requires recalculating node clusters for each new state, computational efficiency is crucial to prevent excessive overhead during training and inference. To address this, we first precompute a set of anchor nodes based on the mesh. During training and inference, each node is assigned to a cluster by referencing the encoding of these anchor nodes, which includes both positional information and current flow field status. This approach reduces computational overhead and stabilizes the patterns encountered by the attention module on the hypergraph, as the anchor nodes remain fixed in position.

- **Comprehensive Evaluation on Benchmark Datasets**: We validate our proposed model on several commonly used benchmark datasets. Additionally, we analyze the performance characteristics of baseline models across datasets with varying features, providing insights and recommendations for further developing more effective CFD surrogate models.

4.2.1 Related Works

In recent years, deep learning has made significant progress in CFD surrogate modeling, leading to the development of various data-driven CFD surrogate models that significantly reduce computational costs while maintaining a certain level of predictive accuracy. Existing deep learning-based CFD surrogate models can generally be categorized based on their specification of the flow field into Eulerian-based and Lagrangian-based approaches [124].

Lagrangian-based surrogate models [69; 72; 129] represent fluid elements as point clouds and learn the evolution of fluid dynamics by summarizing the patterns of these points' motion trajectories and interactions. This approach's key advantage is its suitability for free-surface flows (e.g., multiphase flows, splashing, and water waves) and its ability to handle topological changes such as fluid splitting and merging. However, Lagrangian methods tend to be computationally expensive due to the need to track individual fluid elements and their interactions.

Unlike Lagrangian-based models, Eulerian-based surrogate models [11; 19; 48; 49; 126; 127] describe flow field variables on a fixed grid structure (commonly called a mesh). This specification defines physical quantities such as velocity and pressure on mesh cells or mesh nodes, just like using a network of probes to capture flow field information. Eulerian-based surrogate models remain the dominant choice since most traditional CFD solvers for large-scale engineering applications rely on Eulerian specification for discretization. Our proposed method also falls under this category.

CFD simulations commonly employ irregular meshes to balance computational efficiency and accuracy in key regions of interest (Figure 2, top) [92]. However, earlier deep learning-based CFD methods were often designed for regular grids [126; 127], primarily to leverage well-established machine learning modules such as convolutional neural networks (CNNs). These methods typically interpolate CFD simulation data onto regular grids before training and inference. After prediction, the results must be mapped back to the original irregular mesh again, introducing additional computational overhead and error.

GNNs are naturally well-suited for handling irregular meshes, as these meshes inherently form a graph structure. Early GNN-based applications [92] have already achieved notable success. However, they also highlighted a fundamental challenge in GNN-based CFD modeling: the trade-off between expanding the receptive field to capture long-range dependencies and mitigating oversmoothing. This has led to a series of follow-up studies exploring various techniques to improve the connectivity of mesh-based graphs and reduce the cost of capturing long-range dependencies: Methods based on self-attention mechanisms [11; 49] adjust attention weights for different nodes, allowing more vital information to have priority in propagation. Node clustering-based methods [48; 49] aggregate neighboring nodes into hyper-nodes and apply GNNs on the resulting hyper-node graph to enhance the perception of long-range dependencies. Graph U-Net architectures [19] further sparsify the graph into different coarse-grained hierarchical levels, leveraging a U-Net structure to transmit features across multiple scales. Boundary encoding-based methods [125] directly incorporate the encoding of boundary nodes into each node's feature representation, ensuring that every node is aware of the overall state of the current scenario.

Unfortunately, most of these methods rely on static graph optimization. Two main factors typically drive this choice. First, the computational cost of graph optimization might be too high to make online updates feasible. This highlights the necessity of dynamic graph optimization strategies remaining computationally efficient. Second, in many cases, the objectives of graph optimization are not related to the predictive goals of the CFD surrogate model, making it unnecessary to perform updates during training. For example, when using distance-based K-means clustering, the result remains fixed for a given mesh. This reveals a key potential advantage of dynamic graph optimization: We can learn to adapt the graph structure to contribute to the performance of the downstream task.

4.2.2 Methodology

4.2.2.1 Preliminaries. A Eulerian-based CFD simulation dataset $D = \{R_i \mid i = 1, 2, ..., n\}$ is a collection of simulation runs R, where each R is typically simulated under different global parameters and initial conditions. Each $R = \{T_j \mid j = 1, 2, ..., t\}$ is further composed of multiple time steps T, which express the status of the flow field with fixed time intervals. Each $T = (F, A)$ can be represented as a graph, defined by node features F and edges A. The node features are defined as a matrix $F \in \mathbb{R}^{N \times (C+V)}$, where N is the number of vertices of the mesh, C is the spatial coordinate dimension of each node (typically 2 or 3), and V represents the dimensions of physical variables (such as velocity components, pressure, etc.). The edges are defined as a sparse adjacency matrix $A \in \mathbb{R}^{E \times 2}$, which specifies the indices of the start and end nodes for all E directed edges in the graph. A is typically derived by bidirectionalizing the edges in the mesh.

Given any T'_{init} as an initial time step (T'_{init} typically never appeared as time steps in D), the Eulerian-based CFD surrogate model is designed to predict the flow field in the next time step T'_1, and further iteratively using the prediction result of T'_k to predict T'_{k+1}. It is worth noting that, although it seems to be a time sequence prediction task, most studies treat CFD surrogate modeling as a Markov forecasting task of order one because T'_k contains all the necessary information to predict T'_{k+1}.

4.2.2.2 Framework. In this section, we introduce the overall workflow of FlowCluster and highlight its main contributions. FlowCluster belongs to the node clustering-based CFD surrogate models category, whose general pipeline is illustrated in Figure 4.6.

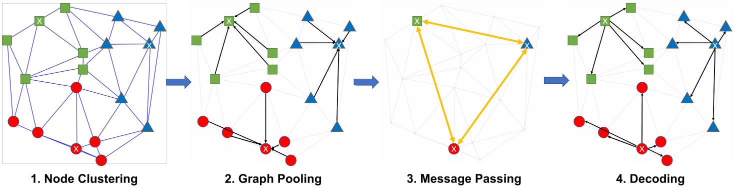


Figure 4.6: The overall workflow of FlowCluster includes four main steps: Node Clustering, Graph Pooling, Message Passing, and Decoding.

Adaptive Node Clustering based on Anchor Nodes. The first step in the workflow is node clustering, where all nodes in a given time step are grouped into clusters. FlowCluster's node clustering process consists of two stages: an offline stage and an online stage.

The offline stage takes only the mesh as input and is independent of the CFD surrogate model's prediction task, allowing it to be pre-computed before training begins. The goal at this stage is to select M anchor nodes from the total N nodes, which will serve as the clustering centers during the online stage. Inspired by the EAGLE model [49], we also adopt the same-size K-means algorithm, clustering the N nodes into M groups based on their spatial coordinates. The node closest to each cluster center is selected as an anchor node.

The online stage leverages both the anchor nodes and the current flow field state to dynamically adjust each node's cluster assignment during training and inference. Specifically, we encode the node feature matrix of each time step T_k using a multi-layer perceptron (MLP). Then, we compute the Euclidean distance between the encoding of each node and the encoding of M anchor nodes, assigning each node to the cluster corresponding to its nearest anchor node.

This two-stage clustering strategy offers several key benefits:

1. **Stable Foundation for Online Encoding**: By fixing anchor nodes, we provide a stable reference for the encoding in the online stage. Since

we aim to discover fine-grained dependencies, we impose minimal constraints on the online encoder. Fixed anchor nodes prevent the formation of imbalanced clusters, which helps maintain stable patterns for downstream modules.

2. **Preserving the Density Distribution of the Original Mesh**: In the offline stage, the same-size K-means clustering ensures each cluster has a similar number of nodes. As a result, the density distribution of the selected anchor nodes closely reflects that of the original mesh. Since the anchor nodes serve as the foundation for cluster assignments in the online stage, the online clustering results retain the characteristics of the original mesh, ensuring detailed modeling at critical locations.

3. **Efficient Online Computation**: In the online stage, clustering only requires encoding the N node features and performing $N \times M$ distance comparisons. The computational overhead remains low since M is significantly smaller than N. This efficiency ensures that the online clustering process can be affordably updated alongside training and inference.

Attention-based Graph Pooling. The second step in the workflow is graph pooling, where the information from all nodes within each cluster is aggregated to form a hyper-node. The graph of these hyper-nodes serves as the input for the next stage. Common choices include mean pooling, max pooling, GRU-based pooling, etc. Given the characteristics of our clustering method, we ultimately opted for a self-attention-based pooling module.

Several reasons led us to this decision. First, we ruled out max pooling, as we were concerned it would discard too much fine-grained information within clusters. Mean pooling was the next to be eliminated because, in our model, a node's cluster assignment is determined by its spatial position and physical features, meaning the selected anchor node is not necessarily close to the cluster center by position, nor is the cluster shape necessarily convex. Since multiple clusters could have spatially overlapping centers, mean pooling could lead to undesirable effects. GRU-based pooling was rejected because GRU outputs depend on the input sequence order. Given that our cluster nodes dynamically change with the evolving flow field and exhibit high randomness, GRU-based pooling could introduce instability. Instead, we opted for another sequence-to-sequence approach: a self-attention-based pooling module.

Figure 4.7 illustrates our self-attention pooling module, essentially a Transformer Encoder with a classification token (CLS token), without positional embedding. In a Transformer Encoder, each token determines how much attention to pay to other tokens in the sequence using the self-attention mechanism, integrating information accordingly into its output representation. As a result, each output token encapsulates information from the entire input sequence. The CLS token is a common technique in Transformer-based sequence encoding, where an additional token is introduced, and its corresponding output serves as a compressed representation of the entire sequence.

We use each cluster's anchor node encoding as the CLS token, while the remaining cluster nodes form the rest of the input token sequence. Since we do not use positional embeddings, the order of the input tokens does not affect the output. Finally, we take the output token corresponding to the CLS token as the graph pooling result, the compressed representation of the entire cluster.

After deciding on the graph pooling module, we explored the node encoding strategy further. Each node's encoding concatenates its physical feature and positional encoding, then fed into the graph pooling module. The physical feature encoding is obtained from an MLP encoder, while the positional encoding is derived from the spectral projection-based encoder proposed in the EAGLE model [49]. Notably, EAGLE suggests combining the absolute positional coordinates with the cluster-relative coordinates to construct the node positional encoding. However, based on our experiments, we propose that whether absolute positional coordinates should be included in the pre-pooling node encoding should be determined by case.

Initially, we attempted to design a universal positional encoding strategy that performs well across all selected baseline datasets. However, we found no single approach that consistently improved performance across all cases. Including absolute positional coordinates in the pre-pooling node encoding improved performance on some baseline datasets but deteriorated results on others. We identified that this discrepancy stems from a fundamental difference between two types of datasets: In the first type, all simulation runs use the same mesh.

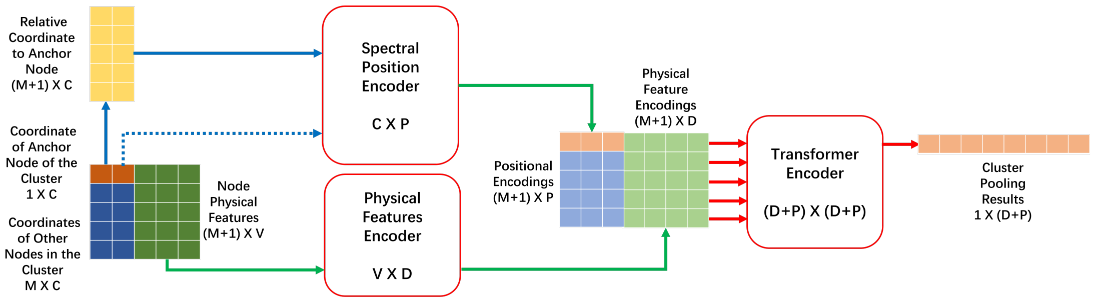

Relative Coordinate to Anchor Node (M+1) X C
Coordinate of Anchor Node of the Cluster 1 X C
Coordinates of Other Nodes in the Cluster M X C
Node Physical Features (M+1) X V
Spectral Position Encoder
C X P
Physical Features Encoder
V X D
Positional Encodings (M+1) X P
Physical Feature Encodings (M+1) X D
Transformer Encoder
(D+P) X (D+P)
Cluster Pooling Results 1 X (D+P)

Figure 4.7: The pooling module is a Transformer Encoder with a CLS token, without positional embedding.

While in the second type, different simulation runs may use different meshes.

Recognizing this distinction helps explain why including absolute positional coordinates has opposite effects in these two cases. Due to the high computational cost of CFD simulations, CFD datasets typically contain only limited simulation runs, making position-specific biases unavoidable. For example, in the ScalarFlow dataset, all simulation runs share the same mesh, and smoke always rises from the center of the ground. Thus, a node located in the corner of the ground and a node in the center of the air exhibit significantly different feature distributions. These differences are embedded in the absolute coordinates of the nodes, naming position-specific biases.

Whether learning position-specific biases is beneficial depends on the surrogate model's intended application. If the model is always deployed in a scenario where the mesh and experimental settings remain similar, then learning these position-specific biases is advantageous for improving predictive performance. However, if the goal is to develop a more generalizable CFD surrogate model, learning such biases leads to overfitting, as the model may be applied to scenarios with different position-specific bias distributions. Both scenarios are common in real-world engineering applications. Therefore, we emphasize that including absolute positional coordinates in the encoding scheme should be considered case by case based on the practical application context.

Message Passing and Decoding. After completing graph pooling, the information from each node cluster is treated as a hyper-node, and message passing is performed on the graph composed of these hyper-nodes. Once message passing is completed, the decoder module reconstructs the hyper-node encodings into predicted values for the individual nodes within each cluster.

Since we found no compelling reason to introduce additional modifications to these two components, we adhered to the principle of minimalism—"do not introduce new entities unless necessary." As a result, we directly adopted the attention-based message-passing mechanism and the GNN-based decoder used in the EAGLE model [49].

4.2.3 Experiments

4.2.3.1 Experimental settings.

Table 4.1: Comparison of Datasets Included in this Study

Name	Solver	Total Runs	Time Steps	Shared Meshes	(Avg.) Nodes
Airfoil [92]	SU2	1200	150	Yes	5233
CylinderFlow [92]	COMSOL	1200	150	No	1885
ScalarFlow [34]	Real-World	104	150	Yes	1000

datasets. All experiments are conducted on three widely used benchmarking datasets: the Airfoil, CylinderFlow, and ScalarFlow datasets. Table 4.1 shows the main characteristics of these datasets. All three datasets are based on irregular 2D triangle meshes. In the Airfoil and ScalarFlow datasets, all simulation runs share the same mesh. The CylinderFlow dataset is more challenging: it uses various meshes across different simulation runs. We truncate all datasets to 150 time steps to maintain consistency with the ScalarFlow dataset.

Baseline Models. We select several models as baselines for comparison. First, we include MGN [92], which represents classical graph message-passing-based CFD surrogate models. Using spline interpolation, GSN [48] predicts with sparsed inputs and then recovers the ignored details. It's included due to its high computational efficiency. Additionally, we consider three models that enhance graph connectivity in different ways. The GATv2 variant [10] of MGN, called MGATv2, incorporate self-attention-based mechanisms. BSMS [19] constructs a Graph U-Net structure, utilizing graphs of different resolutions to improve hierarchical message passing. EAGLE [49] employs static, mesh-based node clustering to enhance graph connectivity while maintaining computational efficiency. For implementation, we make every effort to use the original authors' code. Further details on implementation can be found in the attached code repository.

4.2.3.2 Overall Results. Tables 4.2 and 4.3 present the performance of different models on datasets where the mesh remains the same across all simulation runs (Airfoil and ScalarFlow) and where the mesh varies across different simulation runs (CylinderFlow), respectively. In these experiments, we evaluate two variants of FlowCluster: FlowCluster-A (Absolute), which incorporates absolute coordinate encoding within each cluster, and FlowCluster-R (Relative), which only uses relative coordinate encoding within each cluster. All results are summarized following the evaluation protocol described in section 2.3.2. By analyzing these results, we can draw the following conclusions:

Table 4.2: Overall normalized-RMSE ($\times 1e-2$) of all models on the Airfoil and ScalarFlow datasets. Bold indicates the best performer, underline indicates the second place

Dataset	Airfoil			ScalarFlow		
Rollout	+49	+99	+149	+49	+99	+149
MGN	2.926 ± 0.008	3.392 ± 0.018	3.809 ± 0.044	1.589 ± 0.218	2.765 ± 0.400	3.898 ± 0.488
MGATv2	2.934 ± 0.004	3.398 ± 0.008	3.804 ± 0.010	2.066 ± 0.511	3.477 ± 0.749	4.763 ± 0.921
GSN	3.215 ± 0.016	3.497 ± 0.019	4.088 ± 0.138	2.208 ± 0.313	3.025 ± 0.207	4.511 ± 0.407
BSMS	2.924 ± 0.012	3.380 ± 0.029	3.772 ± 0.056	2.265 ± 0.071	4.100 ± 0.173	5.638 ± 0.272
EAGLE	2.918 ± 0.008	3.365 ± 0.018	3.752 ± 0.030	1.241 ± 0.072	2.354 ± 0.178	3.389 ± 0.345
FlowCluster-A	**2.903 ± 0.017**	**3.343 ± 0.038**	**3.725 ± 0.060**	**1.171 ± 0.120**	**2.117 ± 0.078**	**3.062 ± 0.160**
FlowCluster-R	2.916 ± 0.009	3.372 ± 0.018	3.776 ± 0.026	1.298 ± 0.218	2.365 ± 0.355	3.338 ± 0.408

Table 4.3: Overall normalized-RMSE ($\times 1e-2$) of all models on the CylinderFlow dataset. Bold indicates the best performer, underline indicates the second place

Dataset	CylinderFlow		
Rollout	+49	+99	+149
MGN	4.115 ± 0.014	4.571 ± 0.027	4.773 ± 0.039
MGATv2	4.109 ± 0.014	4.559 ± 0.025	4.756 ± 0.035
GSN	4.475 ± 0.006	4.872 ± 0.002	4.984 ± 0.009
BSMS	4.100 ± 0.025	4.548 ± 0.047	4.743 ± 0.067
EAGLE	4.108 ± 0.018	4.560 ± 0.033	4.757 ± 0.048
FlowCluster-A	4.108 ± 0.010	4.578 ± 0.019	4.792 ± 0.027
FlowCluster-R	**4.044 ± 0.055**	**4.458 ± 0.105**	**4.625 ± 0.146**

1. As discussed in the Graph Pooling section, whether incorporating absolute coordinate encoding is beneficial depends on the application scenario. The results in Table 1 indicate that when the mesh remains unchanged, leveraging absolute coordinate encoding to capture position-

dependent biases improves performance, making FlowCluster-A the best-performing model. However, the results in Table 2 show that modeling these position-dependent biases leads to overfitting when the mesh varies across different simulation runs. In this case, FlowCluster-R achieves the best performance. Overall, FlowCluster demonstrates advantages across all three datasets despite their differing characteristics.

2. A common concern with dynamic graph clustering methods is whether they accumulate errors faster during rollout predictions, leading to instability. Unlike static graph optimization, where errors in the previous time step do not affect the graph structure, dynamic clustering methods introduce an additional source of error propagation. In static methods, predictions are always performed on a fixed graph structure, even if it is suboptimal. In contrast, for dynamic clustering, errors from the previous time step impact the prediction module and influence the clustering process, potentially compounding errors. Indeed, we observe that the standard deviation of performance metrics is slightly more significant for dynamic clustering methods than static ones. However, thanks to rollout training and the inherent advantages of dynamic clustering, FlowCluster remains highly competitive even in long-term rollout predictions.

3. On large-scale datasets, the top-performing models exhibit only slight performance differences, but in terms of absolute performance, all models remain far from practical usability. As we will further illustrate through visualized results in Section 4.2.3.4, a common limitation of existing graph-based CFD methods is that they primarily rely on average RMSE or similar loss functions to supervise training. These metrics do not explicitly ensure the model can accurately capture specific flow structures. We believe that one of the key goals of a better GNN-based CFD surrogate model is to identify different flow structures in the current field. In this regard, dynamic clustering-based methods hold more significant potential than static graph optimization methods.

4.2.3.3 Computational Efficiency. In this section, we compare the computational efficiency of different models, with the results summarized in Ta-

Table 4.4: Comparison of Computational Efficiency (Seconds)

Dataset	Airfoil		CylinderFlow		ScalarFlow	
Scenario	Train	Predict	Train	Predict	Train	Predict
MGN	133	100	48	58	3	10
MGATv2	131	86	58	79	3	15
GSN	28	53	20	35	1	7
BSMS	1829	2870	699	893	23	99
EAGLE	137	183	98	122	5	22
FlowCluster-R	250	221	103	132	6	27

ble 4.4. We report both training time and inference time for each model. Training time refers to the duration required to complete one whole epoch, which means that for each simulation run in the training set, a random time step is selected as the starting point, and the model rolls out forward for 4 time steps during training. Inference time measures the total time required to perform 149 forward-rollout steps for all the simulation runs in the test set.

It is important to note that a model's actual runtime depends not only on computational complexity but also on the efficiency of its implementation. Since we reuse the original authors' code for baseline models as much as possible, there may be some variations in implementation efficiency across different models. Details refer to the code we provided.

The results show that, thanks to our two-stage design, a significant portion of the computational cost is shifted to the offline phase. As a result, despite introducing dynamic node clustering, FlowCluster does not exhibit an unaffordable increase in computational cost compared to the EAGLE model, which serves as our base model.

4.2.3.4 Visualizations. In the first part of this section, we present the visual results of rollout predictions generated by different models as Figure 4.8. We randomly select an initial state from the test set of ScalarFlow, let each model run its prediction, and then interpolate the results onto a regular grid using Cubic Spline Interpolation for visualization. The images are colorized based on the Euclidean norm of the two velocity components, representing the velocity magnitude. For better layout consistency, all results have been rotated by 90 degrees.

By comparing the predictions from different models, we observe that, unfortunately, none can fully capture the complex dynamics of real-world rising smoke on such a dataset with a limited size. However, our model demonstrates a more accurate reconstruction of key characteristics, such as plume height and spread, compared to other baseline models.

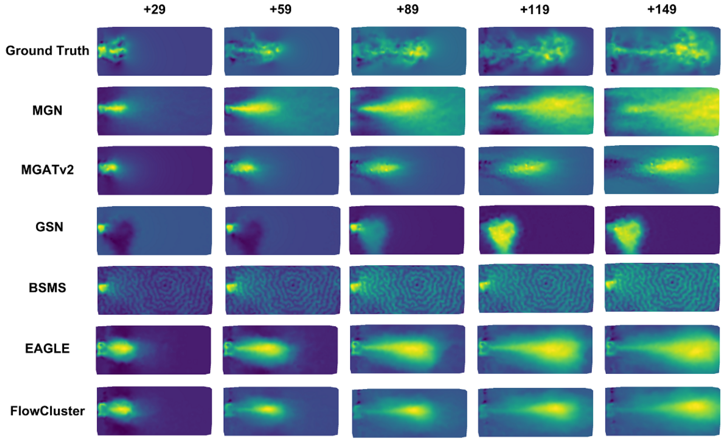


Figure 4.8: Examples of the rollout prediction results of different models on the ScalarFlow dataset.

In the second part of this section, we compare FlowCluster's adaptive node clustering with fixed mesh-based node clustering, as illustrated in Figure 4.9. The data used for this comparison is the same one as in the first part of this section. To improve clarity and emphasize key differences, we crop the lower half of the output for visualization. Within this selected region, we highlight 10 clusters, marking the nodes belonging to each cluster with different colors and symbols while rendering all other nodes in light gray.

The comparison of the two clustering results reveals a key limitation of fixed mesh-based node clustering. Since node clusters remain static across all time steps, a middle-sized flow structure is often split across multiple clusters. Consequently, the decoder must reconstruct these fragmented parts from different hyper-node features and reassemble them into a coherent structure, significantly increasing the decoding complexity.

In contrast, FlowCluster employs adaptive node clustering based on the current flow field state. As the smoke plume rises, the clusters stretch vertically, dynamically grouping nodes based on similar flow characteristics relative to the anchor nodes. This adaptation enhances the internal consistency of each cluster, making it easier for the decoder to reconstruct coherent structures, thus reducing the complexity of the decoding process.

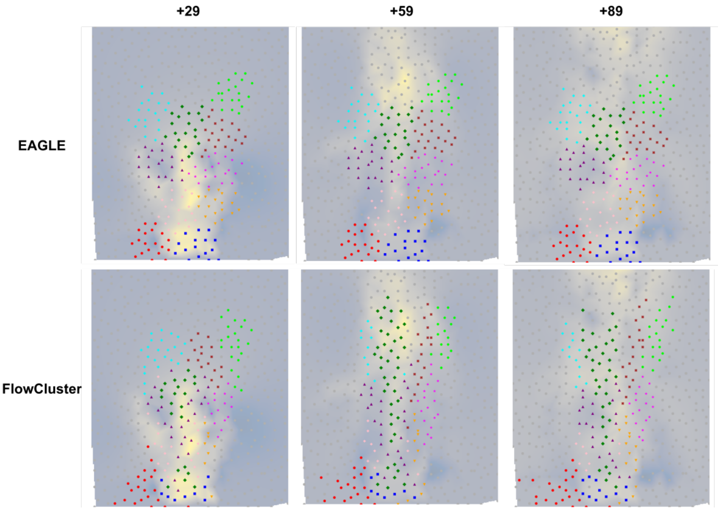


Figure 4.9: Examples of adaptive node clustering from the FlowCluster model are compared with fixed mesh-based node clustering from the Eagle model.

4.2.4 Summary

In this work, we propose a GNN-CFD surrogate model based on Adaptive Graph Refinement to address the limitations of existing GNN-based approaches, such as the over-smoothing problem and reliance on fixed mesh structures. Unlike static optimization methods, our model dynamically refines the graph structure based on the current flow state, improving prediction accuracy by optimally allocating clustering for each node. To mitigate the challenge of computational inefficiency in dynamic optimization, we introduce a two-stage graph clustering mechanism based on pre-selected anchor nodes. Experimental results on benchmark datasets show that FlowCluster shows advantages in long-term iterative forecasting.

4.3 Chapter Summary

We propose two complementary approaches: CFD-based synthetic data generation and GNN-based surrogate modeling for CFD acceleration. The first method leverages Computational Fluid Dynamics (CFD) simulations to generate artificial yet physically consistent aerosol distribution data, supplementing real-world sensor readings. However, given the high computational cost of CFD, the second method introduces a graph neural network (GNN)-based surrogate model, which significantly reduces the computational burden while maintaining the predictive power of CFD-based augmentation. These techniques provide robust data augmentation solutions for addressing data sparsity and imbalance.

5 Temporal Analysis Model

As described in Chapter 2, we used a divide-and-conquer approach to treat temporal and spatial correlations separately. This chapter presents our work on modeling temporal correlations.

5.1 Neural Kernel Network Deep Kernel Learning For Predicting Particulate Matter From Heterogeneous Sensors with Uncertainty

This section proposes a new model pipeline based on a neural kernel network [115] deep kernel learning model. It takes heterogeneous and uncertain data collected from different Internet-connected sources by the "SmartAQnet" [15] as the input. And it predicts the daily average of PM10 concentration readings of the four high-precision PM10 monitoring stations for the next day. As a result, our model pipeline achieved an average mean absolute error (MAE) of 3.67 $\mu g/m^3$ and an average Pearson correlation coefficient (PCC) of 0.665. We also test the effect of different preprocessing strategies and compare our prediction model with other comparison models (baseline, MLP, LSTM, vanilla GPR, etc.). Furthermore, we also validate the contribution of ultra-low-cost sensors in the SmartAQnet sensor network, which reduces the average MAE of our model pipeline from 4.18 $\mu g/m^3$ to 3.67 $\mu g/m^3$ and increases the PCC from 0.589 to 0.665.

5.1.1 Gaussian Process Regression

Gaussian Process Regression (GPR) is a classic non-parametric Bayesian regression algorithm. It has the advantage of performing well on small datasets and provides a measure of uncertainty for predictions.

Unlike many popular supervised machine learning algorithms, such as MLP and LSTM, Bayesian methods don't just learn an exact value for each parameter in a function. They infer the probability distribution of the parameter over all possible values. The way Bayesian methods work is to specify a prior distribution $p(w)$ for the parameter w and then use the Bayesian rule (Equation 5.1) to relocate this probability distribution based on the evidence (that is, observational data).

$$p(w|y,X) = \frac{p(y|X,w) \times P(w)}{p(y|X)} \tag{5.1}$$

The relocated probability distribution $p(w|y,X)$ is called the posterior distribution, containing information from both the prior distribution and the dataset. When we want to predict the label of a point of interest x^*, the predictive distribution can be calculated by weighting all possible predictions by their posterior distribution (Equation 5.2).

$$p(f^*|x^*,y,X) = \int_w p(f^*|x^*,w)p(w|y,X)dw \tag{5.2}$$

Instead of calculating a probability distribution over the parameters of a particular function, GPR calculates a probability distribution over all possible functions that fit the data. In GPR, we first assume a Gaussian process prior, which can be defined by a mean function $m(x)$ and a covariance function $k(x,x')$. The training set and the predicted points of interest are joint multivariate Gaussian distributed from the Gaussian process prior (Equation 5.3). The training process of GPR is to find suitable parameters for the mean and covariance functions. This is usually done with the help of a gradient-based optimizer by maximizing the log marginal likelihood on the training set.

$$\begin{bmatrix} y \\ f^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu \\ \mu^* \end{bmatrix}, \begin{bmatrix} K(X,X)+\sigma_n^2 I & K(X,X^*) \\ K(X^*,X) & K(X^*,X^*) \end{bmatrix} \right) \tag{5.3}$$

5.1.2 Data Description

The data used in this study is freely available on the internet and aggregated by the www.smartaq.net website. For reproducibility, we are using a dataset provided by the SmartAQnet project that covers all measurements from January 1, 2017, to December 31, 2021 [62].

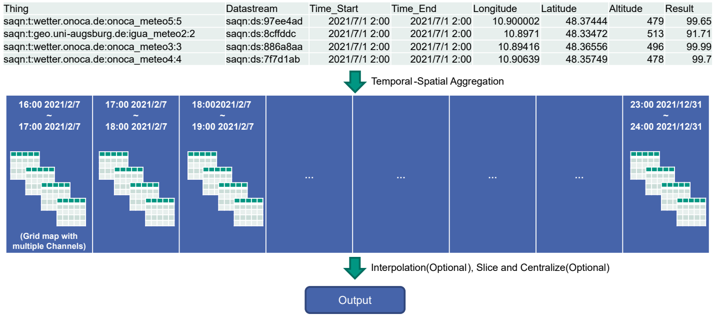
SmartAQnet combines meteorology and aerosol measurement data collected by different entities. A considerable portion of the sensors is located in a rectangular area of 6×4 km that covers most of the city of Augsburg, Germany. The time resolution of the sensor varies by its model. Among them, the high-precision PM10 measuring station generates a record every 1 hour, and the temporal resolution of the vast majority of the other sensors is between 5 minutes and 5 seconds.

In this study, we treat the data from the above-mentioned 4 high-precision PM10 measurement stations as labels. Other data are treated as input data. Several sensors with remote locations and all height profile data are removed. The considered sensors are all located in a rectangular area of about 16×16 km. Among over 30 observed properties provided in the dataset, we select 9 properties that we believe are highly correlated with PM10 concentration to be input into our model. Namely PM10 mass concentration, PM2.5 mass concentration, temperature, relative humidity, air pressure, precipitation, wind direction, wind speed, and global radiation.

5.1.3 Model Pipeline

Current urban air pollutant models can be roughly divided into simulation and statistical models. Among them, the simulation model usually makes predictions by simulating the physical and chemical processes of pollutant diffusion and reaction in the atmosphere [79; 96]. In comparison, the statistical model makes predictions by summarizing the statistical characteristics from historical observations [60; 88; 94; 114]. Our model pipeline adopts a Gaussian process-based statistical modeling approach.

Our model pipeline could be divided into three steps: data preprocessing, feature extraction, and prediction.



Thing	Datastream	Time_Start	Time_End	Longitude	Latitude	Altitude	Result
saqn:t:wetter.onoca.de:onoca_meteo5:5	saqn:ds:97ee4ad	2021/7/1 2:00	2021/7/1 2:00	10.900002	48.37444	479	99.65
saqn:t:geo.uni-augsburg.de:igua_meteo2:2	saqn:ds:8cffddc	2021/7/1 2:00	2021/7/1 2:00	10.8971	48.33472	513	91.71
saqn:t:wetter.onoca.de:onoca_meteo3:3	saqn:ds:886a8aa	2021/7/1 2:00	2021/7/1 2:00	10.89416	48.36556	496	99.99
saqn:t:wetter.onoca.de:onoca_meteo4:4	saqn:ds:7f7d1ab	2021/7/1 2:00	2021/7/1 2:00	10.90639	48.35749	478	99.7

Figure 5.1: The workflow of the Temporal-Spatial Aggregation stage

5.1.3.1 Data Preprocessing. The data preprocessing step is responsible for receiving the readings directly from the sensor network and preliminarily eliminating heterogeneity and uncertainty in the input data through methods such as aggregation and interpolation. The data preprocessing step can be further divided into two stages: temporal-spatial aggregation and window generation.

Temporal-Spatial Aggregation. Data aggregation is a simple and effective way to reduce data uncertainty, especially good at dealing with the influence of symmetrically distributed noise and small probability events. Thanks to the high temporal and spatial resolution of the SmartAQnet network, we can aggregate a considerable number of observation records into one. In this stage, the aggregation will be carried out on all input data on the time and space dimensions (Figure 5.1). All data from the four high-precision PM10 measurement stations, which will be used as labels, are excluded from data aggregation.

In the time dimension, the aggregation resolution is 1 hour, while in the spatial dimension, the spatial extent covered by the dataset (about 16×16 km, as mentioned earlier) is divided into 50×50 grids. That is, The spatial resolution is about 300×300 m. The aggregated data structure is shown in Figure 5.1: the data is represented as several 1-hour time slices in the time dimension. Each time slice is defined as a 50×50 grid with 9 channels to reproduce the spatial relationship of the data. Each channel represents one of the above-mentioned 9 observed properties considered relevant to PM10.

After the aggregation, we consider two additional processing operations:

One processing operation is to slice and center the spatial grid. Precisely, we slice each of the above-mentioned 50×50 spatial grids, ensuring that the position we want to predict is in the center of the sliced grid. We're not sure if the model will benefit from this operation. According to Tobler's First Law of Geography [118], everything is related to everything else, but near things are more related than distant things. By slicing, we hopefully help the model exclude interference from distant noise. On the opposite side, in the SmartAQnet sensor network, some observed properties are only collected by a few sensors. An aggressive slicing may cause the input data to lose too much information about these observed properties, resulting in a dramatic decline in model performance. Therefore, we decided to demonstrate whether slicing should be performed through experimental results.

Another processing operation is to perform interpolation on the spatial grid. We think this helps eliminate heterogeneity in the data further. An obvious benefit is that even the most naive interpolation method can help the model distinguish whether an input 0 is a measured value of 0 or we don't know anything about it. Another potential benefit of interpolation is that it promises to alleviate the shortcomings of the slicing step mentioned above. By interpolation, we can generalize the information of observed properties recorded by only a few sensors to the entire grid, which leads to less information loss due to slicing. In addition, a "wonderful" interpolation that can take land use and the wind into account is expected to improve the homogeneity and expressiveness of the input data significantly. But since this problem belongs to another research direction, and its complexity is no less than the time series prediction problem, we plan to take this topic as a future research direction. In this article we only consider Inverse Distance Weighting (IDW) interpolation.

Window Generation. In this stage, we create time windows using the time slices generated in the previous temporal-spatial aggregation stage. After the window generation stage, a piece of training data for predicting the daily average PM10 concentration on day T looks as shown in Figure 5.2. It consists of two parts: timestamp and time window.

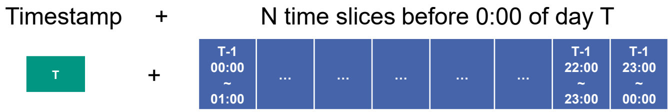


Figure 5.2: A piece of training data processed after the window generation stage for predicting the daily average PM10 concentration on day T

For the timestamp part, we use two-dimensional relative timestamps. One dimension represents how many days have passed from the first day of this year until day T. Another dimension means which weekday day T is. We use relative timestamps because, during GPR training, we observed that GPR is difficult to give effective confidence interval estimates in extrapolation tasks. Because GPR does not find any experience from the training set in the corresponding area, thus it does not have any confidence in its prediction. We transform time series prediction into an interpolation problem using relative timestamps, resulting in more reliable predictions and confidence intervals. But this approach also comes at a cost. For example, it completely ignores the difference between different years. In fact, during the operation of the SmartAQnet project, we experienced the coronavirus pandemic. The lockdown policy is likely to lead to the data pattern over the years are not the same, thus affecting the model performance.

As for the time window part, we select N time slices before 0:00 of the day T, which were generated by the previous temporal-spatial aggregation stage. In our experimental setup, N takes a value of 24. That means the time window includes all the time slices of day T-1.

5.1.3.2 Feature Extraction. The feature extraction step is responsible for receiving the output of the data preprocessing step and performing feature extraction. Feature extraction refers to processing the features that need input to the model through methods such as screening or reorganization to eliminate information redundancy in the input data as much as possible.

In our model pipeline, the feature extraction step mainly has the following two contributions. First, feature extraction reduces the dimension of the input data, making the data more discriminable when the total amount of data is limited. Secondly, during the feature extraction process, the data will lose some unnecessary details (which usually could be treated as noise), which helps further to reduce the impact of data uncertainty on the model.

We tested three feature extraction methods during the model testing phase: Principal Component Analysis (PCA), Convolutional Neural Network (CNN), and Auto-Encoder. Among them, CNN and Auto-Encoder didn't perform well. We believe this is because our dimension reduction task is too heavy (the original input has about 540000 dimensions). At the same time, the total amount of data is too small (only c.a. 1500 available time windows established). PCA, on the other hand, performed well on data generated by all the above-mentioned preprocessing strategies. When set to capture 95% of the variance, PCA can reduce the preprocessed data from 540000 dimensions to $90 \sim 240$ dimensions (depending on the settings of the preprocessing steps). In addition, PCA maintains some data interpretability. We finally decide to use PCA for feature extraction in the model pipeline.

5.1.3.3 Prediction Model. The prediction model is a machine learning model for regression tasks. As mentioned above, our pipeline uses a GPR-based prediction model.

Neural Kernel Network. The covariance function (kernel) is essential to the GP models. The choice of kernel determines almost all the generalization properties of a GP model. This is because the kernel incorporates prior assumptions about the characteristics of the data. In Vanilla GPR, the kernel selection relies heavily on the user's experience and prior knowledge of the data, but this is not always feasible. For example, users' prior knowledge of urban air quality prediction tasks is limited. Moreover, the data is often unrecognizable after many preprocessing steps. In addition, with the rise of combination kernel methods [32; 52; 56; 115], it has been found that adding or multiplying multiple kernels can express more complex priors, which further increases the difficulty of choosing a proper kernel. In this context, the concept of compositional kernel learning is introduced. Its idea is to automate the selection and combination of kernels as part of the training process.

The Neural Kernel Network (NKN) [115] is a compositional kernel learning method. It uses a neural network-styled structure to represent the weighted addition and multiplication of kernels. And it can adjust the weights in the network through back-propagation to automatically select the structure of the combined kernel. The following Figure 5.3 shows the basic structure of NKN.

First, we need to choose some commonly used kernels (such as RBF kernel, linear kernel, periodic kernel, RQ kernel, etc.) as the basic kernels, and these basic kernels are used as the input layer of the network. After that, each network layer can be divided into two parts, the first part is responsible for weighted addition, and the second is for multiplying adjacent results from the weighted addition step with each other. The last layer of the network has only one output, which can be seen as the result of the final combined kernel.

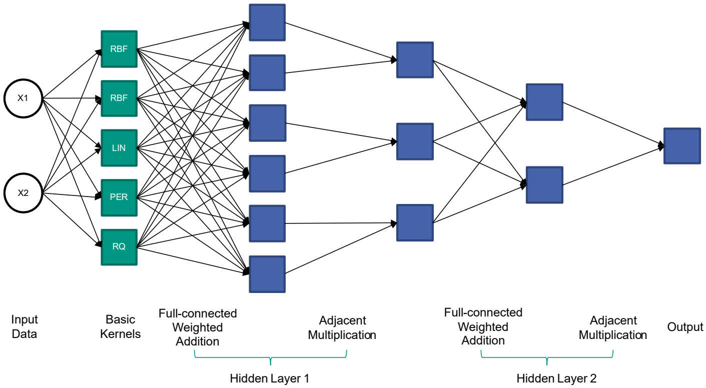


Figure 5.3: The structure of a Neural Kernel Network kernel

Prediction Model Design. In our model pipeline, the prediction model consists of MLP and GPR (Figure 5.4). We first use an MLP with 2 hidden layer to remap the output from the feature extraction step. Then the data will be input to a GPR model with a constant mean function and an NKN kernel for the regression task. We use RBF kernels, RQ kernels, Cosine kernels, and Matern kernels as the basic kernels of the NKN kernel. In the network part of the NKN kernel, we use two layers: the first layer has 4 outputs and the second layer has

1 output. In addition to this structure, we also implement some other regression models for performance comparison. Detailed information will be discussed in the following subsection 5.1.4.

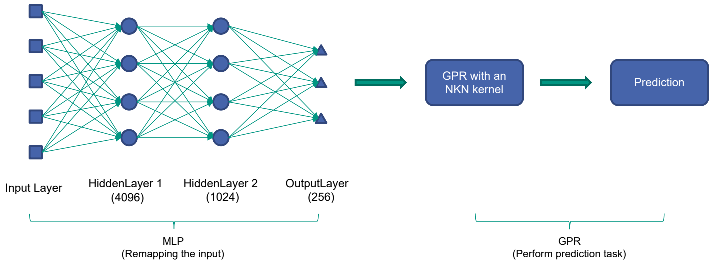


Figure 5.4: The structure of the prediction model

5.1.4 Experiments

5.1.4.1 Model Performance and Comparison. We conducted controlled experiments to determine the optimal model pipeline steps and evaluate our predictive models' performance. As described in Section 3, three components need to be tested and assessed: whether interpolation is required, if grid slicing and centering are necessary, and which predictive model to use. For interpolation, we only consider two cases of no interpolation and IDW interpolation. For grid slicing and centering, we consider three cases: no slicing, big slicing (21×21 grid centered on the predicted point), and small slicing (11×11 grid centered on the predicted point). For the prediction model, in addition to the MLP + NKN kernel GPR proposed above, we also tested another five cases, namely MLP, LSTM, RBF kernel GPR, NKN kernel GPR, and MLP + RBF kernel GPR. That is, a total of 36 sets of experiments were carried out.

For each set of experiments, we first do a hyperparameter tuning. After that, we train 2 times for each of the 4 high-precision PM10 measurement stations and then average these 8 training results as the performance of this pipeline setting. We use this metric to decide which pipeline setting is the best for each prediction model. Then we train additional 3 times with the best setting of each

Table 5.1: The best performance of each prediction model

Model	Best MAE($\mu g/m^3$)	Best PCC	Best Settings	
			Interpolation	**Slice & Center**
Baseline	4.71	0.502	-	-
LSTM	4.04 ± 0.07	0.575 ± 0.020	True	Big Slicing
MLP + RBF_GPR	3.89 ± 0.08	0.624 ± 0.015	True	Big Slicing
RBF_GPR	3.87 ± 0.06	0.635 ± 0.007	True	Big Slicing
NKN_GPR	3.80 ± 0.04	0.641 ± 0.006	True	Big Slicing
MLP + NKN_GPR	3.67 ± 0.08	0.665 ± 0.014	True	Big Slicing
MLP	3.66 ± 0.04	0.690 ± 0.001	True	Big Slicing

model. We compared these results with each other and also with the baseline (using the previous day label as the predicted value). The result of the best performance of each prediction model and in which pipeline settings it was achieved is shown in the following Table 5.1.

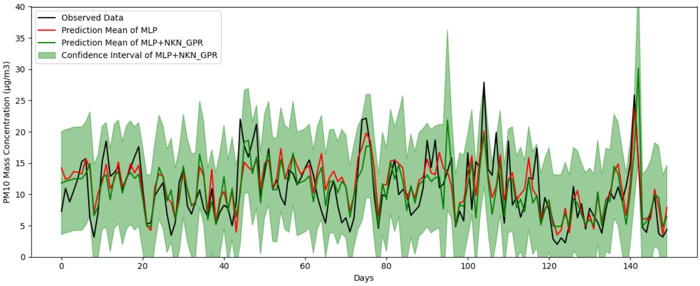


Figure 5.5: MLP model (red line) can only give a single-valued prediction, GPR-based model (green line and interval), however, can also give a reasonable confidence interval of prediction through Bayesian theory.

From the results in Table 5.1, we can draw the following conclusions:

1. All prediction models give the best results in the experimental setting of interpolation + big slicing, which is consistent with our intuition described in subsubsection 5.1.3.1.

2. Among all GPR-based models, the MLP + NKN_GPR model is the best-performing one. Overall, MLP achieves the best performance, and sur-

prisingly, LSTM, which is usually considered a better solution to the time series problem, shows the worst result. After analysis, we believe this should be because the recurrent neural network (RNN) design determines that it is better at dealing with short-term dependencies. As an improvement to RNN, although LSTM has gained the ability to deal with long-term dependencies by adding gates mechanisms such as forget gates, the number of hidden units still limits its ability to express long-term memory. On the other hand, we believe that in this specific problem of PM10 forecasting, the short-term dependencies (such as the distribution over the last hour and the distribution of the same day in the previous week) and ultra-long-term periodic dependencies (such as the same day of the other years) are dominant. In contrast, the more recent long-term dependencies (such as distributions from months ago), which LSTMs are good at handling, have relatively little impact on this problem. Furthermore, it must be pointed out again that MLP and LSTM also has the following shortcomings: MLP and LSTM are uninterpretable model and can only give a single-valued prediction, which means they cannot provide a reasonable confidence interval (Figure 5.5, red line). In many scenarios (such as critical decision-making, when people are more reluctant to make mistakes), an uninterpretable single-valued prediction can only provide very limited help. The GPR-based model can give the confidence interval of prediction through Bayesian theory, which means the prediction given by the model is a Gaussian distribution. We can not only obtain the average value of this distribution (as a single-valued prediction) but also the standard deviation of this distribution can also be obtained (as the model's confidence in its predictions) (Figure 5.5, green line and interval).

3. NKN kernel can effectively improve GP models' performance without prior knowledge of the dataset with the help of compositional kernel learning methods. Figure 5.6 shows the prediction results using the NKN kernel, the Matern kernel and the Cosine kernel. Since different kernels represent different prior assumptions about the dataset, their predictions are also entirely different. The NKN kernel can make multiple assump-

tions through its base kernels and then benefit from all these assumptions by learning the composition structure of these base kernels.

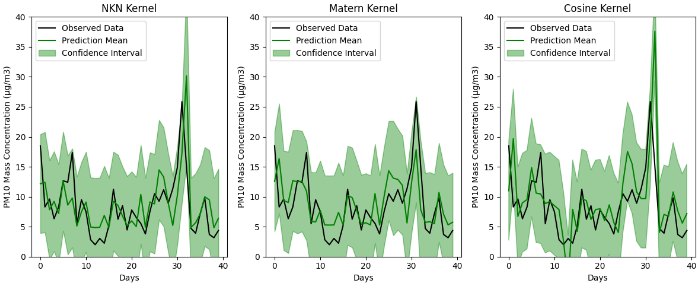


Figure 5.6: Sample of the prediction results using the NKN kernel, the Matern kernel and the Cosine kernel

5.1.4.2 Evaluating the role of low-cost sensors in prediction tasks. As the first machine learning-based time series prediction study on the SmartAQnet dataset, we are also interested in the contribution that low-cost sensors can make to data analysis. Indeed, to improve sensor networks' temporal and spatial resolution, we must make a trade-off in cost. However, there have still been ongoing discussions about whether introducing ultra-low-cost sensors and citizen science projects into the network could benefit the datasets [16]. For this question, we also set up a set of controlled experiments.

The experimental setup is straightforward. We remove all data collected by ultra-low-cost sensors (84 CrowdSensing Nodes) from the dataset and then train the model pipeline with the same experimental setup as the above-mentioned best practice. It is worth noting that after removing these ultra-low-cost sensors, the entire sensor network still has over 100 sensors, which is still very dense for the spatial extent we model. We run each set of experiment 5 times, the following Table 5.2 shows the results of the experiments:

The results of the experiments are evident: although ultra-low-cost sensors introduce more heterogeneity and uncertainty into the aggregated dataset, they can significantly improve the expressiveness of the data when processed and

Table 5.2: The contribution that low-cost sensors can make to data analysis

	MAE ± std ($\mu g/m^3$)	**PCC ± std**
With Ultra-low-cost Sensors	3.67 ± 0.08	0.665 ± 0.014
Without Ultra-low-cost Sensors	4.14 ± 0.04	0.589 ± 0.032
Baseline	4.71	0.502

analyzed appropriately.

5.2 Chapter Summary

This chapter proposes a new model pipeline for time-series prediction of urban particulate matter based on heterogeneous sensor information. The model pipeline takes measurements aggregated from multiple internet sources with high heterogeneity and uncertainty as input and predicts the daily average PM10 mass concentration for the next day. We have experimentally determined the suitable components for the model pipeline: to sequentially perform temporal-spatial aggregation, spatial interpolation, slicing and centering, window generation, PCA dimensionality reduction on the input data, and then use MLP + GP regression with an NKN kernel to perform the prediction task. Ultimately, our model pipeline achieved an average MAE of 3.67 $\mu g/m^3$ and a Pearson correlation coefficient of 0.665.

Furthermore, we experimentally verify the contribution of citizen-run ultra-low-cost sensors in the prediction. Thanks to their meager cost, ultra-low-cost sensors can be large-scale deployed by institutional entities or popularized through citizen science programs. Although these sensors pose challenges such as heterogeneity and uncertainty, they significantly increase the temporal and spatial coverage of the data. We observe that the addition of ultra-low-cost sensor data reduces the average MAE of our prediction model from 4.18 $\mu g/m^3$ to 3.67 $\mu g/m^3$ and increases the PCC from 0.589 to 0.665. As long as processed and analyzed appropriately, ultra-low-cost sensors will definitely result in a significant performance improvement. We thus believe that post-hoc sensor networks that fuse different sensor sources in an opportunistic manner can thus advance knowledge in areas into which classical measurement systems cannot scale due to the cost of initial installation and maintenance. Prediction quality, however, can wildly vary based on the quantity and quality of the local sensor. Thus new prediction methods are needed that can derive certainty measures from the available data. We are confident that Gaussian process modeling can be a key component to inter- and extrapolating heterogeneous information sources. Machine learning approaches like neural networks can greatly help to derive fitting kernels that qualify relations between multiple heterogeneous data sources and choose hyperparameters according to the observed data.

6 Spatial Analysis Model

Following the previous chapter, this chapter presents our advancements in spatial correlation analysis. In fact, throughout our research, we found that the challenges posed by hybrid sensor network datasets have a more significant impact on existing spatial analysis methods than on existing temporal analysis methods. As a result, we have dedicated most of our efforts to developing solutions for spatial correlation analysis. Section 6.2, 6.3, and 6.4 introduce the spatial analysis models we designed to address different challenges, while Section 6.5 discusses the evaluation methods for spatial analysis models based on hybrid sensor networks.

6.1 Related Works

6.1.1 Spatial Interpolation

Spatial interpolation (SI) aims to predict values of a target property at any location (mostly locations without historical observations) according to known observations. Spatial interpolation on sensor network datasets is a vital analysis task in meteorology [24; 43], transportation [145], resource management [73], smart cities [23], etc.

Traditional SI methods, such as Inverse Distance Weighting (IDW) [107] and Spline Interpolation [104], rely on empirical formulas with a limited number of undetermined parameters. These methods struggle to address complex spatial relationships [47]. Consequently, researchers began exploring machine learning-based SI techniques.

A notable category of ML-based SI is Gaussian Process (GP) models, also Kriging [7; 26; 76; 136]. These approaches learn a kernel function that quantifies the correlation between pairs of points. Once this function is learned, it enables the calculation of correlations between the target location and all known locations. However, a significant drawback of GP-based models is that selecting suitable kernel functions necessitates substantial prior knowledge and expertise. Additionally, GPs are very computationally and memory intensive.

Numerous deep-learning models have been developed to analyze spatial data with two prominent families: Graph Neural Networks (GNNs) and Transformers.

Transformer-based models interpret the input as a sequence of tokens. They adaptively extract the correlation between input tokens with the multi-head self-attention mechanism. However, we also note that existing transformer-based spatial interpolation models still use dense input that takes all known observations of the same location as a token, benefiting from its homogeneity to learn stable representations. For example, Fan et al. [38] put known observations on grid maps and processed them with Vision Transformer, Yu et al. [141] removes all sensing stations with more than 25% missing data, and Feng et al. [39] interpolates the missing data with linear interpolation. However, we believe that Transformer could also treat sparse observations as variable-length token sequences and, therefore, be highly compatible with the heterogeneity of HWSN datasets.

Recently, GNN-based models have emerged as a prominent approach. In GNN-based spatial interpolation methods, observed locations are represented as nodes, while spatial relationships (e.g., distance and azimuth) define edges, forming a graph. GNNs learn shared patterns for propagating and aggregating information across the graph, enabling predictions at unobserved locations. Generic GNN architectures [41; 122] have already shown promising results in spatial interpolation tasks, sparking researchers' interest in refining GNN models to address the unique challenges of this field. For instance, models like PE-GNN [58] and LSPE [33] enhance positional encoding to better represent location-related features, while approaches such as KCN [3] and SPONGE [85] focus on improving the encoding of spatial correlations between nodes.

However, most existing GNN-based spatial interpolation models still rely on simple heuristics for graph construction, such as fully connected graphs [132], K-nearest neighbors [42; 58; 85; 133], threshold-based approaches [30;

112; 64; 139], or natural relations [5; 135]. These methods often fall short when complex factors beyond straightforward spatial correlations influence the spatial distribution of the target variable. In such cases, nodes may require multi-hop message passing to access truly correlated neighbors. Yet, GNNs are susceptible to over-smoothing [98], a phenomenon where node embeddings become indistinguishably similar as the number of layers increases. These challenges highlight the necessity of introducing GSL into spatial interpolation tasks to construct graphs that better reflect task-relevant spatial correlations.

6.1.2 Graph Structure Learning

GNN is highly sensitive to the input graph structure [148]. The primary goal of GSL is to generate a graph structure better suited to the downstream task. GSL methods can be divided into two subcategories: graph structure construction and graph structure refinement. The main difference is that graph structure construction learns the graph structure from scratch based solely on the node feature matrix. In contrast, graph structure refinement uses an initial adjacency matrix as the basis. Our model falls under the graph structure construction.

From the perspective of how edges are generated, GSL models can be divided into metric-based, neural, and direct methods [147]. Metric-based methods [68; 140; 143; 146] use kernel functions to generate edge features from corresponding node pair features. Neural methods [10; 33; 77] adopt deep learning architectures (such as MLPs, attention mechanisms, or transformers) to learn edge features from node pair features. Direct methods [123; 134] treat the adjacency matrix as a learnable parameter. Still, because optimizing the adjacency matrix and the downstream GNN simultaneously usually involves non-differentiable operations, these methods typically rely on iter-training to update both components at different stages. Our proposed model belongs to the neural methods.

From the training perspective, GSL models can be categorized into co-training-based, pre-training-based, and iter-training-based models [144]. Co-training [131; 140] optimizes the GSL module alongside the downstream GNN, using the GNN's task performance as a supervision signal. Pre-training [67; 75] first trains the GSL module using self-supervised losses, after which the GSL module is fixed, and then the downstream GNN is trained. Iter-training [74; 150] alternates between fixing the GSL while training the GNN to convergence and then fixing the GNN while training the GSL to convergence. This paper follows the co-training-based approach, which we prefer for several reasons. First, co-training optimizes the GSL according to the task performance, ensuring high task relevance. Second, pre-training requires an additional metric to evaluate what constitutes a "good" graph structure and when to stop pre-training. For instance, in node classification tasks, people believe nodes with similar features should be connected or share similar local structures. Since most GSL research focuses on classification tasks, there is limited empirical evidence on which self-supervised signals are suitable for spatial interpolation tasks. Our experience shows that introducing such metrics will add inductive bias to the model. Unless this inductive bias consistently aligns with the task's requirements, it can lead to performance degradation on datasets that do not fit it. Thus, we would like to avoid introducing unreliable inductive biases into our model. Finally, iter-training requires multiple rounds of convergence, significantly increasing the needed computational resources. Given that sensor networks generate observation frames in high temporal resolution, the size of the datasets grows rapidly as deployment time increases, so it's important to control the computational cost of the model.

6.2 Isolating Latent Context Information Enhances Graph Structure Learning for Spatial Interpolation

Graph Structure Learning (GSL) has emerged as a promising approach for constructing graph structures reflecting task-relevant correlations. However, applying existing GSL methods to spatial interpolation tasks often fails to deliver the anticipated performance improvements. This limitation stems from the nature of real-world datasets, where the spatial correlations of target variables are influenced not only by observed features but also by unobserved factors. These unobserved influences may arise from omissions during sensor network design, restricted access to proprietary data, sporadic events, etc. The information on the influence of observed features on the target variable is termed

Spatial Correlation Information (SCI), which the GSL module can effectively leverage and safely generalize. Conversely, the portion shaped by unobserved factors is called Latent Context Information (LCI). Misattributing LCI to observed features undermines the generalizability of the GSL module, degrading model performance. Thus, isolating the impact of LCI during GSL training is essential to ensuring robust and generalizable graph structures.

It is challenging to isolate LCI completely, but this work does not aim for that. We propose the Information Segmentation Spatial Interpolation (ISSI) Model, which focuses on isolating two types of LCI to enhance model performance.

The first type, Random Noise LCI, is characterized by low occurrence probabilities or symmetric distributions, making its effects mitigable through input smoothing. Examples of Random Noise LCI include sensor noise and rare sporadic events. To address this, we introduce a Transformer-based Global Information Fusion module that corrects each location's encoding (i.e., node feature) by referencing all other node features. This not only smooths inputs but also incorporates global context into each node feature, allowing the inner-product kernel-based GSL module to learn better graph structures.

The second type, Location-specific LCI, relates to attributes specific to individual locations, like land use or differences in sensor quality across locations. Location-specific LCI is especially addressed because the spatial coverage rate of sensor network datasets is usually low, resulting in a high bias in Location-specific LCI. This causes the GSL module to learn distorted spatial correlations, which reduces the generalizability of the GSL module and degrades the model performance. To address this, we design a two-branch self-supervised GSL module. The main branch processes original inputs, while the centered branch uses relative coordinates to the target location, obscuring Location-specific LCI by mixing location-specific attributes across locations. Maximizing agreement between the graph structures generated from the two branches achieves two goals: (1) reducing Location-specific LCI bias in graph construction and (2) approximating Location-specific LCI via residuals between node embeddings of the two branches. This approximation further refines predictions in downstream tasks.

Our contributions can be summarized as follows:

- We propose the Information Segmentation Spatial Interpolation (ISSI) model, a GNN-based framework incorporating a self-supervised GSL module. Unlike existing methods that rely on simple coordinate-based heuristics, our approach leverages GSL to better represent task-relevant spatial correlations, leading to significant performance gains.

- The ISSI model introduces mechanisms to mitigate the negative impacts of Random Noise LCI and Location-specific LCI on the GSL module. By effectively isolating these influences, our model constructs more robust graphs with higher generalizability, further enhancing model performance.

- We evaluate our model on diverse, publicly available real-world datasets spanning different application scenarios. Experimental results demonstrate that the ISSI Model consistently outperforms baseline methods, showcasing strong adaptability and universality across tasks with varying characteristics.

6.2.1 Task Definition

GSL-based spatial interpolation tasks can be formally expressed as follows:

Given N observed locations, represented by a matrix $M \in \mathbb{R}^{N\times(T+S+C)}$. Where T is the dimension corresponding to the target variable (usually 1), S is the dimension of other variables that may affect the spatial distribution of the target variable (like wind and humidity can affect the air pollutants), and C represents the dimension of spatial coordinates (usually 2 or 3).

Arbitrarily specify a location $P \in \mathbb{R}^{1\times C}$, add P as a virtual node to M (missing information is filled with zeros). This will result in a node feature matrix $X \in \mathbb{R}^{(N+1)\times(T+S+C)}$. Our model, consisting of GSL and GNN modules, uses X as input. The GSL module generates an adjacency matrix A according to X: $A = GSL(X), A \in \mathbb{R}^{(N+1)\times(N+1)}$, and the GNN module subsequently performs information transfer according to X and A: $O = GNN(X,A), O \in \mathbb{R}^{(N+1)\times T}$. The interpolation result is $O(P) \in \mathbb{R}^{1\times T}$, which means the GNN output that corresponds to the virtual node P.

6.2.2 Framework

6.2.2.1 Global Information Fusion. We use an inner-product kernel-based GSL method. Unlike neural approaches that process node pairs individually or direct methods requiring alternating optimization alongside downstream tasks, the inner-product kernel reduces computational demands for generating adjacency matrices by learning node feature embeddings and calculating their inner products. It can also provide stable inductive bias as constraints.

However, this method has a notable limitation: each value in the adjacency matrix is determined solely by the embeddings of the corresponding node pair without considering the overall graph context. This can result in suboptimal graph structures. We designed a Transformer-based Global Information Fusion Module to address this issue, integrating global context into each node's embedding before the GSL module learns the adjacency matrix. Specifically, we treat each node's embedding as an input token and process the sequence of tokens using two vanilla Transformers without positional encoding. Their output sequences will be used to calculate the inner product. With the multi-head attention mechanism, each output token of the Transformer adaptively captures correlations from all other tokens. This integrates global context into each node embedding.

Additionally, the Global Information Fusion Module smooths the node feature matrix. Due to its sporadic nature or symmetric distribution, the effects of random noise LCI tend to neutralize when viewed across the entire dataset. The Global Information Fusion Module, trained on the dataset, learns a generalized mechanism to encode each node by referencing all other nodes. When noise presents in the input tokens during prediction, the module leverages the learned mechanism to correct each token, effectively smoothing the input and mitigating the influence of Random Noise LCI.

6.2.2.2 Approximates the Location-specific LCI. Another type of LCI that requires isolation is Location-specific LCI, which presents a significant challenge due to the lack of explicit indicators identifying which parts of the input belong to it. To address this, we designed a self-supervised two-branch framework to approximate Location-specific LCI.

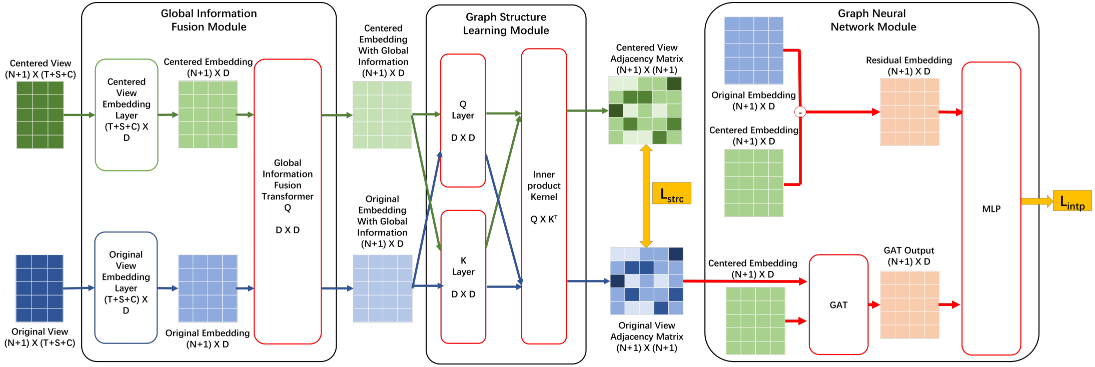


Figure 6.1: The ISSI model comprises global information fusion, two-branch self-supervised GSL, and GNN. The embeddings, learnable modules, and data flows in the Centered and Original branches are marked in green and blue, respectively. Those shared by both branches are red, and the loss signals are yellow.

The first branch, referred to as the Original Branch, takes the original node feature matrix X as input, which we term the Original View. In the Original View, a specific coordinate c corresponds to one specific location, including the associated Location-specific LCI, e.g., on a grassland. During training, the modules in the Original Branch learn to fit all information in the input, including both SCI and Location-specific LCI.

The second branch, referred to as the Centered Branch, takes the Centered View X' as input, derived from X by replacing the absolute coordinates with relative coordinates to the target location:

$$X' = \{(t_i, s_i, c_i - c_P)_{i=1}^{i=N+1}\} \in \mathbb{R}^{(N+1)\times(T+S+C)} \tag{6.1}$$

, where t_i, s_i, and c_i are the target variable, support variable, and absolute coordinate of the i-th observation in X, respectively, and c_P is the absolute coordinate of the target position P. In the Centered View, using relative coordinates intentionally removes the specific information associated with absolute coordinates. A coordinate c in the Centered View no longer corresponds to a fixed location. Its actual location varies with the target location. For example, in the Centered View, the coordinate might correspond to a grassland for one target location or a forest for another. With sufficient target locations in the training set, we can assume that the patterns learned from the Centered View fit SCI and the dataset-wide average of Location-specific LCI.

We use separate embedding layers to encode the Original View and Centered View, generating the Original Embedding and Centered Embedding, respectively. These embeddings are then passed through a shared GSL module to produce their corresponding adjacency matrices, AM_{ori} and AM_{cen}. The agreement between these matrices is maximized by minimizing the loss: $L_{strc} = MAE(AM_{ori}, AM_{cen})$. Due to the Location-specific LCI partially missing in the Centered Branch, a deterministic GSL module cannot produce multiple distinct graph structures from the same input. Consequently, the portion of the information that the GSL module cannot explain corresponds to the difference between Location-specific LCI and its dataset-wide average. We treat this difference as a reasonable approximation of Location-specific LCI representation.

The GSL module produces two outputs for downstream tasks: (1) a reasonable approximation of Location-specific LCI, termed the Residual Embedding, which is the difference between the Original Embedding and the Centered Embedding, and (2) the adjacency matrix AM_{ori} from the Original Branch.

6.2.2.3 Graph Neural Network Module. The GNN module is responsible for completing the downstream spatial interpolation task. We chose GraphSAGE [41] as the GNN module since it shows the best performance among vanilla GNN models, but other information transfer-based GNN models, such as GCN [57] and GAT [122], are also compatible here. It uses Residual Embedding as the node feature matrix and fully connected AM_{ori} as the adjacency matrix.

After the GNN module performs information transfer, we get the GNN output $O(P)$ corresponding to the virtual node P. Then we calculate the MAE loss between the prediction $O(P)$ and label $L(P)$ as the interpolation loss: $L_{intp} = MAE(O(P), L(P))$. We use the linear combination of L_{strc} and L_{intp} as the final loss signal during training: $L = L_{intp} + L_{strc}$

6.2.3 Experiments

6.2.3.1 Experimental Setup.

Table 6.1: Comparison of Datasets Included in this Study

Name	Sensor Type	Noise Level	Observed Channels	Average Nodes	Spatial Coverage Rate[1]
SAQN	All fixed-location	High	9	45.8	12.30%
ABO	All movable	Low	3	154.0	4.68%
Marine	mixed	Pass quality inspection	6	61.5	97.96%

1. How many grids have been observed at least once in the entire dataset

Datasets. We evaluate our approach on three publicly accessible real-world datasets: the SmartAQnet dataset (SAQN) [63], the NOAA Aircraft-Based Observation dataset (ABO) [86], and the Copernicus In-situ Marine Observation dataset (Marine) [105]. Detailed information is provided in Table 6.1.

The SAQN dataset originates from a fixed-location sensor network monitoring urban air quality and meteorological variables. Substantial noise levels characterize it due to the widespread use of low-cost sensors and limited spatial coverage owing to its reliance on fixed-location sensors. Consequently, this dataset exhibits high Random Noise LCI and Location-specific LCI bias. The ABO and Marine datasets demonstrate the growing trend of integrating mobile sensors into sensor networks, which improves spatial coverage but results in complex sensor topologies. The ABO dataset employs sensors mounted on commercial aircraft to measure meteorological parameters. While notable for its low noise levels, it features fewer observed channels and limited spatial coverage (as a three-dimensional dataset), meaning it primarily reflects Location-specific LCI bias. On the other hand, the Marine dataset combines low noise levels with high spatial coverage, making it an ideal benchmark for assessing the GSL performance under low LCI conditions.

Baselines. To ensure a comprehensive comparison, we include GraphSAGE [41] as it serves as the foundational component of our model. From non-GSL-based GNN spatial interpolation models, we evaluate KSAGE [3] and PE-SAGE [58]. From GSL-based spatial interpolation models, we evaluate LSPE [33] and SPONGE [85]. Additionally, we incorporate SSIN [65] and SMACNP [6] as representative non-GNN-based approaches.

6.2.3.2 Overall Performance. Table 6.2 shows the performance of all above-mentioned models on all datasets. After random searches on hyperparameters, each model was trained 20 times on each dataset, with four-fold leave-one-area-out cross-validations and five random seeds (1, 2, 3, 4, and 5). The evaluation metrics are Mean Absolute Error (MAE) and R^2, commonly used in spatial interpolation tasks. All results are summarized following the evaluation protocol described in section 2.3.3. By comparing the results, we answer the following questions:

Q1: Which is the overall best-performing model?

A1: Across all three datasets, the ISSI model consistently demonstrates the best performance. Notably, none of the other baselines exhibit a comparable ability to maintain stable performance across diverse tasks. On the ABO dataset, the second and third best-performing models are SMACNP and PE-SAGE, respectively. SSIN and PE-SAGE take these positions on the Marine

Table 6.2: Overall Result of all models. Bold indicates the best performer, underline indicates the second place.

Model	ABO		SAQN		Marine	
Metrics	MAE(°C)	R^2	MAE($\mu g/m^3$)	R^2	MAE(°C)	R^2
GraphSAGE	10.293 ± 0.044	0.451 ± 0.004	5.863 ± 0.048	0.317 ± 0.009	1.993 ± 0.038	0.631 ± 0.012
KSAGE	14.268 ± 0.021	0.012 ± 0.002	5.535 ± 0.048	0.301 ± 0.010	3.128 ± 0.018	0.198 ± 0.007
LSPE	13.844 ± 0.424	-0.390 ± 0.409	6.205 ± 0.115	0.171 ± 0.030	1.660 ± 0.084	0.721 ± 0.026
PE-SAGE	3.302 ± 0.258	0.927 ± 0.008	6.115 ± 0.243	0.217 ± 0.047	1.315 ± 0.042	0.835 ± 0.011
SSIN	18.800 ± 0.469	-0.420 ± 0.062	6.197 ± 0.084	0.167 ± 0.034	1.035 ± 0.052	0.893 ± 0.014
SPONGE	3.918 ± 0.296	0.913 ± 0.013	6.388 ± 0.138	0.249 ± 0.019	1.593 ± 0.071	0.768 ± 0.023
SMACNP	3.241 ± 0.281	0.884 ± 0.025	6.237 ± 0.337	0.201 ± 0.062	1.741 ± 0.044	0.287 ± 0.127
ISSI	**2.021 ± 0.025**	**0.974 ± 0.001**	**5.385 ± 0.095**	**0.382 ± 0.022**	**0.944 ± 0.019**	**0.911 ± 0.004**

dataset. However, these models suffer significant performance degradation on the SAQN dataset. KSAGE and GraphSAGE emerge as the second and third best-performing models on SAQN. In summary, ISSI stands out as the top-performing model.

Q2: Compared with heuristics-based graph structures, is GSL more helpful for spatial interpolation on sensor network data?

A2: Yes, but it is crucial to consider the characteristics of sensor network datasets. The results indicate that generic GSL methods cannot consistently guarantee performance improvements and, in some cases, underperform compared to simple heuristic-based graphs. This is because, while heuristic-based graphs are not always optimal, they at least provide a stable inductive bias. In contrast, sensor network datasets often interfere with the GSL module's generalization ability. The GSL module may generate graph structures worse than those constructed using simple heuristics, negatively impacting downstream performance. Our model addresses this issue by isolating the influence of LCI. As a result, our model demonstrates better average performance, stability, and cross-task adaptability than existing methods.

We also noticed that the more variate the sensor topology, the more pronounced the performance improvement provided by ISSI compared to its basic GraphSAGE model. It is only 8.2% on the SAQN dataset, 47.4% on the Marine dataset, and as high as 80.4% on the ABO dataset. This shows that GSL can better address the challenges of the variate topology of movable sensor networks by learning the general pattern of building graph structures.

Q3: Except for the performance advantage of ISSI, are there other signs that reveal the existence of LCI and the need to isolate it?

A3: The performance of PE-SAGE across different datasets inspired this research. The difference between PE-SAGE and GraphSAGE lies in adding a learned absolute coordinates-based embedding to node features. This effectively improves the performance of PE-SAGE on the Marine dataset with only a slight decline in stability. However, on the ABO and SAQN datasets, the standard deviation of MAE is approximately five times larger than that of GraphSAGE. This indicates that the absolute coordinates-based embedding tends to overfit certain location-specific factors, which, as we now understand, stem from the high bias in Location-specific LCI of these two datasets. In contrast, our ISSI model does not exhibit such behavior. It consistently achieves stable performance gain compared to its GraphSAGE backbone across all three datasets.

Q4: Why do many models degrade performance on the SAQN and ABO datasets?

A4: First, the SAQN and ABO datasets show low spatial coverage, resulting in high location-related biases. Models that employ learnable location-based encodings (e.g., PE-SAGE, LSPE, SPONGE, SSIN) are particularly susceptible to these biases, leading to performance degradation. Second, the SAQN dataset has a high noise level. Heavy parameterized models, when not constrained by denoising approaches, tend to overfit the noise, resulting in volatile performances. Third, the target variable of the ABO dataset (air temperature) has an evident stratification along the altitude dimension. However, models like GraphSAGE, KSAGE, and SSIN use Euclidean distance-based encoding for spatial correlations, and unlike PE-SAGE and CESI, they do not incorporate additional location-based embeddings. The hidden inductive bias is the spatial isotropy of the Euclidean distance, which deviates far from the truth. Our model, on the contrary, successfully overcomes all these challenges.

6.2.3.3 Ablation Study. We use the ablation study to determine the impact of different modules on final performance. We implemented three ablation models: (1). ISSI w/o GI: this ablation model removes the Transformer-based global information fusion from the ISSI model. That is to say, we no longer integrate global information into each node feature. Instead, we obtain the node features through two single-layer fully connected layers. (2). ISSI w/o SS: this ablation model removes the centered branch used for self-supervised learning from the ISSI model. That is, we no longer try to isolate Location-specific LCI. Instead, like most GSL methods, the input node features are directly used to produce graph structures and downstream GNN modules. (3). ISSI Null: this ablation model simultaneously removes the global information fusion Transformer and self-supervised learning branch. Ablation models are trained according to the same settings and standards as above. Table 6.3 shows

Table 6.3: Result of Ablation Study. Bold indicates the best performer, underline indicates the second place

Model	ABO		SAQN		Marine	
Metrics	MAE(°C)	R^2	MAE($\mu g/m^3$)	R^2	MAE(°C)	R^2
ISSI	2.021 ± 0.025	**0.974 ± 0.001**	**5.385 ± 0.095**	0.302 ± 0.022	0.944 ± 0.019	0.911 ± 0.004
ISSI w/o GI	**2.004 ± 0.015**	**0.974 ± 0.001**	5.635 ± 0.093	0.284 ± 0.016	0.970 ± 0.024	0.906 ± 0.003
ISSI w/o SS	2.498 ± 0.013	0.961 ± 0.001	5.507 ± 0.181	0.308 ± 0.019	**0.934 ± 0.018**	**0.912 ± 0.002**
ISSI Null	2.473 ± 0.018	0.962 ± 0.001	5.461 ± 0.181	**0.311 ± 0.025**	0.970 ± 0.028	0.911 ± 0.006

the results of the ablation study.

Due to its high spatial coverage, the Marine dataset doesn't contain much bias introduced by Location-specific LCI. As a result, the self-supervised branch does not improve performance. The main performance gains on this dataset come from the Global Information Fusion module. Models incorporating global information fusion (ISSI and ISSI w/o SS) achieve lower average MAE and reduce the standard deviation by approximately 30%.

On the ABO dataset, the low spatial coverage leads to pronounced bias on Location-specific LCI. Consequently, we observe that models with the self-supervised branch (ISSI and ISSI w/o GI) achieve around 20% improvement in average MAE.

Both high Location-specific LCI bias and high Random Noise LCI characterize the SAQN dataset. In this case, using only one module results in performance degradation due to misattributions. In contrast, the model performs better when both modules are used together. Although the average MAE improvement is modest, it is noteworthy that models with the self-supervised branch (ISSI and ISSI w/o GI) reduce the MAE standard deviation by approximately 50%, highlighting its ability to mitigate the impact of Location-specific LCI bias and stabilize the model's performance.

The results demonstrate that our designed modules function as intended, effectively improving the model's average performance and stability.

6.2.3.4 Comparing the Learned Graph Structure. In this section, we compare the graph structure learned by ISSI with the one based on K-nearest neighbors (KNN, k=5) and inverse distance weighting, commonly used in current GNN-based spatial interpolation models. This comparison highlights the differences between these approaches.

We randomly selected three data entries from the Marine dataset and plotted all nodes according to their coordinates, with virtual nodes marked with 'C' on the right. Graph structures were generated using the original data entries and then propagated a pseudo-signal through these graph structures. In this pseudo-signal, the node features of virtual nodes were set to 1, while all other nodes were set to 0. After propagation, nodes were assigned different colors based

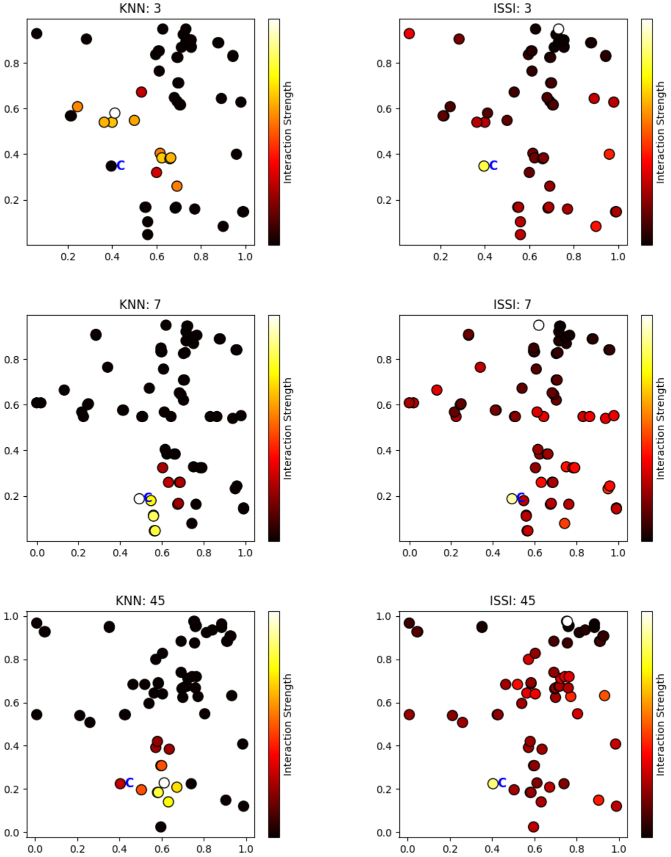


Figure 6.2: Comparing the ISSI learned graph structure with KNN-based heuristics

on the final values of their features, visually representing each node's "interaction strength" over others, indicating where and how much its information is disseminated. Darker colors represent weaker interaction strength.

The comparison results, shown in Figure 6.2, reveal clear distinctions. In the KNN-based graph, nodes exhibit firm control over their closest neighbors but minor influence on distant nodes. In contrast, the graph structure learned by ISSI demonstrates an extensive interaction range, with smoothly varying intensity and occasional jump connections across long distances. This property makes the ISSI-learned graph structure better suited for spatial interpolation tasks, as it effectively captures long-distance dependencies while reducing susceptibility to bias, occasional failures, and noise from individual sensors.

6.2.4 Summary

This paper proposes the ISSI model, a spatial interpolation framework incorporating a self-supervised GSL module. The self-supervised mechanism approximates and isolates bias in Location-specific LCI during GSL training. This helps the GSL module learn more generalizable graph structures, particularly for sensor network datasets where Location-specific LCI bias is often pronounced due to low spatial coverage. By effectively mitigating these biases, our model safely harnesses the capabilities of the GSL module to construct better graph structures for downstream interpolation tasks adaptively, thereby achieving improved performance. Additionally, our GSL module leverages the Inner-product Kernel with a global information fusion step to provide a stable inductive bias and maintain computational efficiency. Experimental results demonstrate that the ISSI model delivers consistent and robust performance across real-world datasets with diverse characteristics and application scenarios, underscoring its broad applicability and potential for practical deployment.

6.3 Feature Deviation Embedding Improves Graph Structure Learning for Spatial Interpolation

Applying existing GSL methods to spatial interpolation tasks doesn't always improve performance. The GSL module takes each node's location and ob-

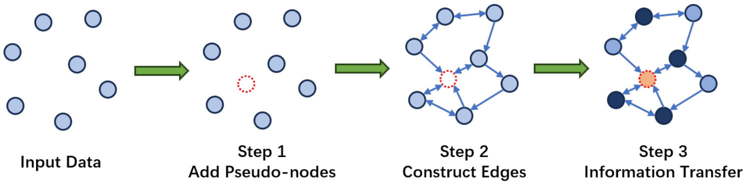


Figure 6.3: Workflow of GSL-based spatial interpolation.

served values as input. In sensor network datasets, both of them are systematically imbalanced. Real-world observed variables often approximate imbalanced distributions, such as Gaussian or Gamma distribution. The interactions and combinations of different observed variables make this problem even more complicated. The imbalance in node locations might arise naturally. For instance, datasets based on crowd sensing are influenced by population density. Sometimes, node location imbalance is even intentional. For example, due to budget constraints, sensor deployment often sacrifices density in less essential regions to prioritize critical areas.

When facing a feature-imbalanced dataset, the model will be biased towards the majority features and thus underestimate the importance of the minority features [59]. Therefore, the generalization ability of the model might be negatively affected. Imbalances in node locations can lead the GSL module to overemphasize frequently occurring spatial patterns, skewing the model's ability to infer spatial correlations. Similarly, imbalances in observed values can cause the GSL module to favor often occurring values, potentially ignoring uncommon but critical observations.

To address the above problems, we propose the Feature Deviation Embedding Graph Spatial Interpolation (FDE-GSI) model. The model aims to correct the GSL module by explicitly modeling how the deviation of node features affects the result. However, complex node feature distributions and noises in the training set usually cause this modeling to lack generalization ability. Therefore, we seek an embedding scheme to filter and map the essential node features to a high-dimensional latent space and make the latent features approximate a multi-dimensional Gaussian distribution. Such latent distribution with regular trends makes modeling with higher generalization ability possible. We designed an adversarial optimization structure based on an adjustable information bottleneck to optimize this embedding scheme adaptively.

We train two encoders to encode the node locations and observed values to the latent space and conclude their feature deviations, respectively. They are called Feature Deviation Encoders (FDEs). The task of the FDEs is to map all node features into a high-dimensional latent space and maximize their total probability density on the standard Gaussian distribution. As the FDE optimizes, the embeddings of different node features become increasingly similar, which means the bottleneck progressively narrows. When the total probability density reaches maximum, all node features are mapped to the mean vector of the standard Gaussian distribution and thus lose all their distinctions, which means the bottleneck is entirely closed. Meanwhile, the GSL and GNN modules use the FDE outputs to predict interpolation results. A wider bottleneck benefits their performance by allowing FDEs to retain more details of the node features. By jointly optimizing these components, the model learns how to encode the most important features by adversarially finding the minimum necessary bottleneck.

In summary, our main contributions are as follows:

- We propose FDE-GSI, a GSL-based spatial interpolation model. Its GSL module corrects the impact of imbalanced feature distributions by explicitly perceiving feature deviations embedded by encoders trained with adversarial optimization structures based on adjustable information bottlenecks.

- We are one of the few that introduce GSL into spatial interpolation tasks. Furthermore, to our knowledge, we are the first to point out that feature-level imbalances systematically exist in GSL for spatial interpolation tasks and propose corresponding solutions.

- We conducted experiments on several real-world datasets with different characteristics. The results demonstrate that our model outperforms the best existing models across all datasets, confirming its good performance and strong task adaptability.

6.3.1 Preliminaries

From the GSL perspective, the spatial interpolation task can be formally defined as follows:

The model takes two inputs. The first input is the known readings at N different locations, which can be represented as a matrix $M \in \mathbb{R}^{N\times(T+C+S)}$. Among them, T represents the dimension of the target variable (usually 1), C represents the dimension of the spatial coordinates (usually 2 or 3), and S represents the dimension of other variables that can affect the spatial distribution of the target variable (for example, wind direction can significantly affect the spatial distribution of air pollutants, hereinafter referred to as support variables). The second input of the model is the spatial coordinates of an arbitrary target location $P \in \mathbb{R}^{1\times C}$ (most P never appear in the training set).

Use the pseudo node generator to supplement the values on T and S dimensions for P, and add this resulting pseudo node to the matrix M to obtain a new matrix $X \in \mathbb{R}^{(N+1)\times(T+S+C)}$ (Figure 6.3 Step 1). X can be regarded as a node feature matrix containing $N+1$ nodes. The GSL module takes X as input and returns an adjacency matrix A (Figure 6.3 Step 2):

$$A = GSL(X), A \in \mathbb{R}^{(N+1)\times(N+1)} \tag{6.2}$$

The GNN module subsequently performs information transfer using X and A (Figure 6.3 Step 3):

$$O = GNN(X,A), O \in \mathbb{R}^{(N+1)\times T} \tag{6.3}$$

The interpolation result is $O(P) \in \mathbb{R}^{1\times T}$, which means the GNN output that corresponds to the pseudo node of P.

6.3.2 Methodology

In this section, we introduce the proposed FDE-GSI model, whose structure is illustrated in Figure 6.4. The model can be divided into three stages: the Data Preparation Stage, the GSL Stage, and the GNN Stage.

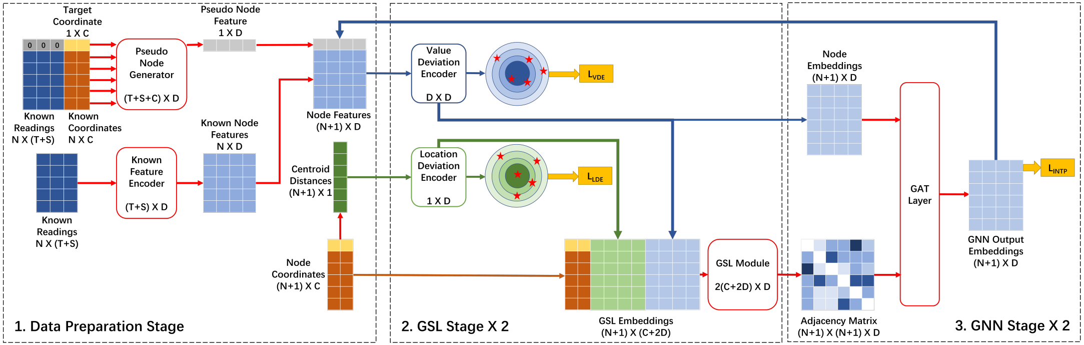

Target Coordinate 1 X C
0
0
0
Known Readings N X (T+S)
Known Coordinates N X C
Pseudo Node Generator (T+S+C) X D
Pseudo Node Feature 1 X D
Known Readings N X (T+S)
Known Feature Encoder (T+S) X D
Known Node Features N X D
Node Features (N+1) X D
Centroid Distances (N+1) X 1
Node Coordinates (N+1) X C
1. Data Preparation Stage
Value Deviation Encoder D X D
L_{VDE}
Location Deviation Encoder 1 X D
L_{LDE}
GSL Embeddings (N+1) X (C+2D)
GSL Module 2(C+2D) X D
2. GSL Stage X 2
Node Embeddings (N+1) X D
Adjacency Matrix (N+1) X (N+1) X D
GAT Layer
GNN Output Embeddings (N+1) X D
L_{INTP}
3. GNN Stage X 2

Figure 6.4: Overview of proposed FDE-GSI model.

6.3.2.1 Data Preparation Stage. In the Data Preparation Stage, we process the input data into embeddings required by the GSL module. As introduced above, the input of the model includes the Target Coordinate (marked in yellow in Figure 6.4), a series of Known Readings (marked in dark blue), and their corresponding Known Coordinates (marked in red). The embeddings required by the GSL module consist of Node Coordinates, Centroid Distances, and Node Features.

Getting Node Coordinates and Centroid Distances is simple. Node Coordinates are obtained by concatenating the Target Coordinate with the Known Coordinates. To get Centroid Distances, we calculate the centroid of the Known Coordinates (i.e., the average value of each dimension) and then compute the Euclidean distance from each node's coordinates to this centroid. We regard the centroid distance as a representation of a node's remoteness. The larger the centroid distance, the more remote the node is in the entire graph.

Node Features are formed by concatenating the Pseudo Node Feature and Known Node Features. The Known Node Features are generated by processing the Known Readings through a multi-layer perceptron (MLP). Generating the Pseudo Node Feature for the target location, which can be any arbitrary location, is a unique challenge in GNN-based spatial interpolation, as the sensor network typically lacks corresponding observed values. Existing methods include padding with zeros or simple interpolation techniques (e.g., linear interpolation or inverse distance weighting). We employ a Transformer-based pseudo-node feature generator. As shown in the upper left corner of Figure 6.4, we first form a matrix of $(N+1) \times (T+S+C)$ dimensions by zero padding the pseudo node. This matrix can be viewed as a sequence of $(N+1)$ tokens, each with $(T+S+C)$ dimensions. We feed this sequence into a Transformer Encoder, and its output corresponding to the pseudo-node token is used as the Pseudo Node Feature.

6.3.2.2 GSL Stage. The primary task of the GSL Stage is to generate the corresponding graph structure from the input, including the adjacency matrix and associated edge features. Our GSL Stage also refines the Node Features, producing Node Embeddings that serve as the input to the downstream GNN module.

In the GSL Stage, the primary focus is addressing the challenge of feature-level imbalance. Sensor network datasets systematically exhibit imbalances in observed values and node coordinates, which can distort the decision boundaries of models by causing them to overly prioritize frequently occurring features. Specifically, imbalances in node locations lead the GSL module to overemphasize common spatial relationships in training data, narrowing the boundaries for interpreting other spatial patterns. This results in learned spatial patterns excessively not being smooth. Similarly, imbalances in observed values can cause the GSL module to over-prioritize frequently occurring values, which might incorrectly downplay the influence of nodes that are more critical to the target but less frequent in the training data. These degrade the generalization ability of the GSL module and generate suboptimal graph structures for downstream GNNs, ultimately impacting the performance of the spatial interpolation model.

Our fundamental idea is to incorporate the feature deviation as part of the GSL module's input, allowing the explicit modeling of the distortions caused by imbalanced features. Thus, the challenge turns to measuring and encoding feature deviations. In other words, what base distribution do we assume when talking about "deviation"?

We propose that the assumed distribution be as smooth and regular as possible, which helps the model to extrapolate safely. However, this distribution must not deviate too far from the actual feature distribution, as an unrealistic assumption also impedes the safe generalization of learned patterns. Complicating this further, the training sets provided by spatial interpolation tasks are often poor samples of the actual feature distribution. Feature distributions in training sets tend to be imbalanced and irregular, making inferring the actual feature distribution difficult.

However, upon observation, we found that when several nodes are clustered closely, their observed values often approximate each other, which reminds us of Tobler's first law of geography [117]. This led us to realize that sensor network datasets usually contain redundant information. The asymmetry between the little information gained from this redundancy and the distortion caused by its resulting feature imbalance gave us an important insight: filtering out some redundant information could force the input features to conform better to a specified regular distribution. In other words, rather than attempting to infer the actual feature distribution, we filter and reshape the input features to match a predefined target distribution.

Based on this idea, we designed two Feature Deviation Encoders (FDEs) to filter and embed information from node features and centroid distances. These are called the Value Deviation Encoder (VDE) and Location Deviation Encoder (LDE). Aside from their input dimensions, the structures of VDE and LDE are identical. Each FDE is an MLP that maps the input into a D-dimensional latent space. To optimize this filtering and encoding process, we introduce an adjustable information bottleneck, using adversarial optimization to learn how to balance the retention and filtering of input features.

Specifically, we maximize the total probability density of the FDE's output z for all nodes on a D-dimensional standard Gaussian distribution $p(x) \sim \mathcal{N}(0_D, \mathbf{I}_D)$. This is equal to minimizing the encoding loss L_{FDE} (also named L_{VDE} and L_{LDE} since there are two FDE with different inputs, as shown in Figure 6.4):

$$L_{FDE} = \sum_{i=1}^{N} \left(p\left(\mathbf{0}_D\right) - p\left(\mathbf{z}_i\right)\right) \tag{6.4}$$

As FDE is optimized, the feature deviation embeddings for different inputs become increasingly similar, which signifies that the information bottleneck is narrowing. In the extreme case, when the total probability density reaches its maximum, all input features are mapped to the mean vector of the Gaussian distribution. This indicates the complete closing of the bottleneck, and all features lost their distinctions.

Since the FDE's output generates the Node Embeddings and Adjacency Matrix, the downstream GNN Stage will attempt to predict the correct interpolation results using compressed information from the bottleneck. Obviously, the GNN Stage prefers a wider information bottleneck because it allows the FDE to retain more details from the original input. The model can adaptively find the minimal necessary bottleneck through adversarial optimization between these two modules, thus encoding the most critical features as latent embeddings that approximate a standard Gaussian distribution.

The final component of the GSL Stage is a neural-based GSL module. Specifically, this module is an MLP that takes the GSL Embeddings of every pair of nodes as input. It outputs edge features for all possible edges, thus forming a fully connected graph. Then, we remove self-loops in the adjacency matrix, resulting in the final graph structure.

6.3.2.3 GNN Stage.

In the GNN Stage, the GNN module performs information transfer based on the Node Embeddings and Adjacency Matrix generated from the GSL Stage, resulting in GNN Output Embeddings. The GNN Stage comprises a single GATv2 layer [10].

Our model contains two rounds of information transfer, each independently learning its graph structure. As illustrated in Figure 6.4, the GSL and GNN stages have two independent copies to handle the first and second transfer separately. After the first transfer, the resulting GNN Output Embeddings replace the initial Node Features generated during the Data Preparation Stage, becoming the input to the second GSL Stage. After the second transfer, the GNN Output Embeddings corresponding to the pseudo node are extracted and passed through an MLP decoder, which produces the final interpolation result and computes the MAE, namely interpolation loss L_{INTP}.

The total loss is the sum of the interpolation loss and the four encoding losses derived from the two GSL Stages:

$$\begin{aligned} L = L_{INTP} + L_{VDE1} + L_{LDE1} \\ + L_{VDE2} + L_{LDE2} \end{aligned} \tag{6.5}$$

6.3.3 Experiments

6.3.3.1 Experimental Setup.

Datasets. We conducted experiments on three publicly available real-world datasets: the SmartAQnet dataset (SAQN) [63], the NOAA Aircraft-Based Observation dataset (ABO) [86], and the Copernicus In-situ Marine Observation dataset (Marine) [105]. Table 6.4 summarizes the key characteristics of these datasets.

Table 6.4: Comparison of Datasets Included in this Study

Name	Target OP	Sensor Type	Noise Level	Spatial Dimensions	Spatial Coverage Rate[1]	Average Nodes
SAQN	PM10	All fixed	High	2	12.30%	45.8
Marine	water temperature	mixed	Pass quality inspection	2	97.96%	61.5
ABO	air temperature	All movable	Low	3	2.11%	57.0

[1] This percentage means how many pixels have been observed at least once in the entire dataset.

The SAQN dataset is a typical example of the traditional fixed-location sensor network dataset. It is collected entirely from fixed-location sensors, meaning the only network structure changes arise from decommissioning old sensors and adding new ones. This lack of dynamic structural variation in the dataset makes the GSL module more susceptible to biases introduced by imbalances in node locations. Additionally, the dataset contains higher noise levels due to the widespread use of low-cost sensors.

In contrast, the ABO and Marine datasets represent a recent trend in sensor network datasets: the inclusion of moveable sensors. The Marine dataset combines observations from fixed-location (buoys) and moveable (ships) sensors. The ABO dataset is exclusively composed of moveable sensor data, with observations recorded by airborne sensors installed on commercial aircraft. On the one hand, moveable sensors introduce more significant variability in the potential graph structure, mitigating the impact of node location imbalance by providing more spatial patterns in the training set. On the other hand, the spatial relationships in the test set are also more random and complex, which poses a more significant challenge to the generalization ability of the GSL module. In addition, the ABO dataset is also unique in that it is a dataset with three-dimensional spatial coordinates, and its spatial correlation is not isotropic (the lower atmosphere has prominent stratification characteristics in the vertical direction but not in the horizontal direction).

Baselines. We selected the following baseline models for comparison. First, we included three classic GNN models: GCN [57], GAT [122], and Graph-

SAGE [41]. Then, we included the PE-GNN [58] from non-GSL-based GNN spatial interpolation models. Among GSL-based methods, we included the KCN [3] and NodeFormer [131]. The basis model of our FDE-GSI (without VDE and LDE) is also included as a vanilla neural GSL-based approach, which we named NN-GSL. Finally, we include the non-GNN-based spatial interpolation model SSIN [65].

For GCN, GAT, and GraphSAGE, we used implementations available in Pytorch-Geometric. For the other baseline models, we utilized the implementations provided by their respective authors. The appendix provides further details about implementation, environments, and other training settings. When the paper is accepted, we will release all codes on GitHub.

We also considered iter-training-based GSL models, such as SUBLIME [75], but ultimately failed to include them in our results. The primary reason was that such models require multiple rounds of convergence. Given that each dataset used in this study contains millions of graphs available for training, the computational resources needed for iter-training-based models far exceeded what was feasible within our ability, which we also think is a drawback of such models.

6.3.3.2 Overall Performance. Each model was trained 20 times on each dataset, with four-fold leave-one-area-out cross-validations and five random seeds (1, 2, 3, 4, and 5). Table 6.5 shows the overall performance. All results are summarized following the evaluation protocol described in section 2.3.3. By comparing the results, we can answer the following questions:

Q1: Who is the overall best performer?

A1: Our proposed FDE-GSI model ranks first in all three datasets' MAE and R^2. On the SAQN dataset, our model shows a 6.94% improvement in average MAE compared to the second-best model, NodeFormer. On the Marine dataset, our model achieves an 11.30% improvement in average MAE compared to the second-best, SSIN, with a notable 38.46% reduction in MAE standard deviation, indicating greater consistency across different random seeds. Furthermore, on the ABO dataset, our model demonstrates an 18.58% improvement in average MAE over the second-best, PE-GNN, with a remarkable 86.47% reduction in MAE standard deviation.

Table 6.5: Overall Result of all models. Bold indicates the best performer, underline indicates the second place

Model	SAQN		Marine		ABO	
Metrics	MAE($\mu g/m^3$)	R^2	MAE($^\circ$C)	R^2	MAE($^\circ$C)	R^2
GCN	5.875 ± 0.048	0.264 ± 0.011	3.575 ± 0.113	-0.412 ± 0.099	10.298 ± 0.490	0.149 ± 0.064
GAT	5.495 ± 0.074	0.348 ± 0.008	2.416 ± 0.045	0.484 ± 0.019	9.409 ± 0.081	0.213 ± 0.011
GraphSAGE	5.863 ± 0.048	0.317 ± 0.009	1.993 ± 0.038	0.631 ± 0.012	9.114 ± 0.267	0.261 ± 0.028
PE-GNN	6.115 ± 0.243	0.217 ± 0.047	1.315 ± 0.042	0.835 ± 0.011	2.864 ± 0.207	0.873 ± 0.009
KCN	5.535 ± 0.048	0.301 ± 0.010	3.128 ± 0.018	0.198 ± 0.007	10.396 ± 0.066	0.087 ± 0.007
NodeFormer	5.430 ± 0.050	0.378 ± 0.010	2.168 ± 0.096	0.604 ± 0.027	12.190 ± 1.682	-0.033 ± 0.234
NN-GSL	6.240 ± 0.361	0.103 ± 0.098	1.058 ± 0.076	0.881 ± 0.032	4.237 ± 2.583	0.709 ± 0.308
SSIN	6.197 ± 0.084	0.167 ± 0.034	1.035 ± 0.052	0.893 ± 0.014	12.325 ± 0.161	-0.156 ± 0.025
FDE-GSI	**5.053 ± 0.059**	**0.406 ± 0.012**	**0.918 ± 0.032**	**0.920 ± 0.008**	**2.332 ± 0.028**	**0.936 ± 0.004**

Interestingly, the second-best model differs across the three datasets: PE-GNN ranks seventh, fourth, and second, respectively; NodeFormer ranks second, sixth, and eighth; and SSIN ranks eighth, second, and ninth. This variability highlights that no baseline model consistently performs well across all datasets, whereas our FDE-GSI model maintains superior and stable performance across datasets with different characteristics. Overall, this analysis confirms that our proposed model is the best performer.

Q2: How do existing GSL-based methods perform?

A2: Overall, existing GSL-based methods show inconsistent performance across different spatial interpolation tasks. For instance, KCN ranks fourth, eighth, and seventh across the three datasets; NodeFormer ranks second, sixth, and eighth; and NN-GSL ranks ninth, third, and third. This suggests these GSL methods sometimes fail to learn generalizable graph structures for spatial interpolation tasks.

Given the significant difference between different GSL methods, we focus on comparing NN-GSL and FDE-GSI, as our FDE-GSI is a direct improvement over NN-GSL. On the Marine dataset, where data quality and spatial coverage rate are high, FDE-GSI improves NN-GSL's average MAE by 13.23% and the MAE standard deviation by 57.89%. As data quality and spatial coverage rate decrease, the improvements become more pronounced. On the SAQN dataset, FDE-GSI shows a 19.02% improvement in average MAE and an 83.66% improvement in MAE standard deviation over NN-GSL. On the ABO dataset, despite better data quality than the SAQN dataset, the meager spatial coverage rate leads to even more significant improvements, with 44.96% and 98.92% gains in average MAE and MAE standard deviation, respectively. This analysis supports the conclusion that feature imbalance in the dataset is a critical factor affecting the performance of GSL-based spatial interpolation methods.

Q3: Why do many models perform particularly poorly on the ABO dataset?

A3: There are at least two reasons. The first one is this dataset's poor spatial coverage. This issue significantly impacts models with weaker inductive biases, such as NodeFormer and SSIN. These Transformer-based models heavily rely on learning patterns from the dataset, so the quality and expressiveness of the dataset significantly influence their performance. The second one is that the dataset violates the spatial isotropy assumption, which many models rely on. The dataset mainly observes the lower atmosphere, where the target variable (air temperature) exhibits clear stratification in the vertical direction but not in the horizontal direction. GCN, GAT, and GraphSAGE use distance-based KNN heuristics to construct graph structures, while KCN uses distance-based kernel functions to learn graph structures. These models inherently assume spatial isotropy, as the distance is spatial isotropic. The conflict between this assumption and the actual data distribution led to catastrophic failures in these models.

6.3.3.3 Ablation Study. We implemented the following ablation models to investigate the contributions of each key component in our proposed approach: (1). NN-GSL: This is the base model without any optimizations and serves as a baseline for comparison. (2). FDE-GSI w/o IBN: In this model, we remove the adaptive information bottleneck from FDE-GSI. (3). FDE-GSI w/o LDE: In this model, we remove the Location Deviation Encoder from FDE-GSI. (4). FDE-GSI w/o VDE: In this model, we remove the Value Deviation Encoder from FDE-GSI.

We trained and evaluated these models using the same setup and evaluation metrics as in the main experiments. The results are presented in Table 6.6. By analyzing the results, we arrived at the following conclusions:

Comparing the performance of FDE-GSI w/o IBN and FDE-GSI reveals that the adaptive information bottleneck is crucial for the model's success. The bottleneck mechanism helps the model learn how to map only the most essential information into a latent variable that follows a multidimensional standard Gaussian distribution. This design prevents the model from being biased by noise or frequently occurring but uninformative features in the dataset. Additionally, regularizing the latent embedding distribution enhances the model's reliability during extrapolation. Consequently, this improves both the generalization ability (as evidenced by the significant increase in average performance) and robustness (as shown by the reduced performance variance) of our GSL module.

Table 6.6: Result of Ablation Study. Bold indicates the best performer, underline indicates the second place

Model	SAQN		Marine		ABO	
Metrics	MAE($\mu g/m^3$)	R^2	MAE($^\circ$C)	R^2	MAE($^\circ$C)	R^2
NN-GSL	6.240 ± 0.361	0.103 ± 0.098	1.058 ± 0.076	0.881 ± 0.032	4.237 ± 2.583	0.709 ± 0.308
FDE-GSI w/o VDE	7.321 ± 2.748	-0.502 ± 1.464	1.400 ± 0.503	0.685 ± 0.265	7.228 ± 5.268	0.270 ± 0.690
FDE-GSI w/o LDE	5.879 ± 0.694	0.225 ± 0.181	0.973 ± 0.078	0.909 ± 0.015	2.518 ± 0.295	0.923 ± 0.019
FDE-GSI w/o IBN	5.933 ± 0.898	0.161 ± 0.307	1.933 ± 0.873	0.541 ± 0.338	4.234 ± 2.857	0.687 ± 0.372
FDE-GSI	**5.053 ± 0.059**	**0.406 ± 0.012**	**0.918 ± 0.032**	**0.920 ± 0.008**	**2.332 ± 0.028**	**0.936 ± 0.004**

From comparing FDE-GSI w/o VDE and FDE-GSI w/o LDE, we observe that the Value Deviation Encoder (VDE) is more critical than the Location Deviation Encoder (LDE) in our model. Models using only the VDE achieved second place on all three datasets. In contrast, using the LDE alone did not improve performance. The LDE's main contribution lies in the fact that when combined with the VDE, it can further enhance the average performance and significantly improve robustness. Therefore, we conclude that imbalances in observed values and node locations are common in real-world datasets, with value imbalances being more dominant. Addressing only one of these imbalances can make the GSL module more prone to overfitting. In lower-quality datasets, overfitting in the GSL module had a more pronounced negative impact on the final results.

Thus, combining all key components has proved essential for stable and reliable spatial interpolation.

6.3.3.4 Comparing the Learned Graph Structures. This section visually compares the graph structures generated by different methods to explain further why our approach is more advantageous for spatial interpolation tasks. We randomly selected several evaluation data from the Marine dataset as a case study. First, we let different models generate graph structures based on the data. Then, we propagate pseudo data, whose central node is set to 1 and all other nodes are set to 0, through the model according to this graph structure. The propagation results reflect how the GNN spreads the information of the central node. We compare graph structures generated by KNN, NN-GSL, and our proposed FDE-GSI. The results are shown in Figure 6.5. In the figure, the central nodes are marked with the letter "C" on the right side, and their values decide the midpoint of the color scale of the corresponding figure.

By observing the results, we come to the following conclusions:

GCN utilizes the traditional distances-based KNN graph structure. This structure exhibits strong locality, meaning that information rarely propagates to distant nodes. One evident issue is that when a node is located too far from others, it can barely propagate its information. When it is used for spatial interpolation, only a few nearby nodes determine the result. On one hand, this

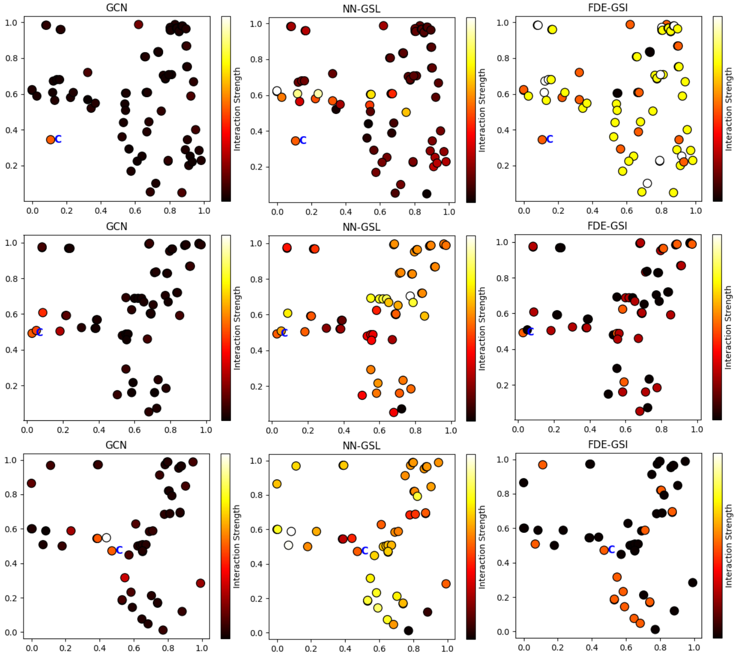


Figure 6.5: Comparing different graph structures

makes it difficult for the model to capture global patterns effectively. On the other hand, if the neighboring nodes are affected by noise, the model's performance will be significantly impacted.

NN-GSL, which represents current GSL-based methods, has already shown notable improvements. The graph structure allows information to propagate to further nodes. Another prominent feature of this structure is that the information propagation varies smoothly in the spatial dimension, with nearby nodes typically sending similar levels of information. While this benefits the model's stability by averaging across many nodes, it may also dilute finer details.

While our FDE-GSI model also allows a wide propagation range, it typically selectively propagates node information rather than simply using a smooth spatial pattern. This reflects the effect of the information bottleneck, where the model selects the most important and reliable information for focused propagation. This balance between global patterns and local details is likely a significant factor in our model's success.

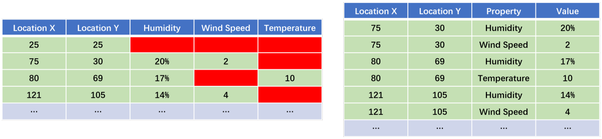
6.3.4 Summary

In this paper, we introduce FDE-GSI, a GSL-based spatial interpolation model. Our GSL module corrects the impact of imbalanced feature distributions by explicitly perceiving feature deviations through encoders trained with adversarial optimized information bottlenecks. We conducted experiments on multiple real-world datasets with varying characteristics, and the results demonstrate that our model outperforms existing state-of-the-art models across all datasets, showcasing its strong performance and adaptability to different tasks. We believe that developing new methods for feature-level imbalance learning holds promise for further enhancing the capabilities of GSL-based models in spatial interpolation tasks.

6.4 CESI: Sparse Input Spatial Interpolation for Heterogeneous and Noisy Hybrid Wireless Sensor Networks

In-situ sensor networks are crucial in many fields, offering high temporal coverage and robustness to interference. Modern sensor networks increasingly combine sensors of varying costs to balance budget and deployment density, enabling finer-grained data collection for more detailed modeling. We hereafter refer to them as hybrid wireless sensor networks (HWSNs) [63; 91].

Low-cost sensors, while economical, often compromise accuracy and reliability, leading to highly heterogeneous and noisy HWSN datasets. This poses significant challenges for spatial interpolation models, which are traditionally designed based on homogeneous, high-quality sensor data [6; 38; 58; 65; 85]. These methods typically require dense, wide-format input (Figure 6.6 left), which is increasingly incompatible with modern IoT protocols like OGC SensorThings API [70] that favor narrow, sparse data formats (Figure 6.6 right) to cope with the heterogeneity of HWSN. Consequently, there is a pressing need for spatial interpolation models tailored to sparse input in HWSNs, which will introduce the following potential benefits:



Location X	Location Y	Humidity	Wind Speed	Temperature
25	25	🟥	🟥	🟥
75	30	20%	2	🟥
80	69	17%	🟥	10
121	105	14%	4	🟥
...	...	...	...	...

Location X	Location Y	Property	Value
75	30	Humidity	20%
75	30	Wind Speed	2
80	69	Humidity	17%
80	69	Temperature	10
121	105	Humidity	14%
121	105	Wind Speed	4
...	...	...	...

Figure 6.6: An example of the wide format (left) and narrow format (right) of the same input data entry of spatial interpolation models.

First, dense input models require extensive imputation to handle missing values in HWSN datasets. While traditional spatial interpolation methods also discuss data imputation, the causes of missing values in HWSNs differ significantly, resulting in much higher imputation workloads. In traditional sensor networks, missing values are mainly caused by occasional sensor failures. With high-quality sensors, such issues are infrequent. Thus, recent spatial interpolation studies still consider simple techniques like linear interpolation [39] or removing incomplete rows/columns [141] acceptable. In contrast, HWSNs face far more frequent failures from low-cost sensors, compounded by heterogeneity in sensor types, where sensors at different locations may only measure subsets of the observed properties. For instance, applying PE-GNN [58] to the SmartAQnet dataset [63] required imputing over 50% of the inputs, with all rows containing missing cells. In such cases, removing incomplete data is infeasible, while excessive imputation alters feature distributions and accumulates errors, degrading model performance. By using sparse input, we can prevent the data imputation step and the above-mentioned disadvantages. When converting dense input to sparse input (see Figure 6.6 as an example), we can safely delete the readings with missing values because this will not cause collateral damage to other readings as it will be for dense input.

Second, dense input models typically encode all properties at the same location (a row in Figure 1 left) and focus on location-level correlations. In contrast, sparse input models encode each observation (a row in Figure 1 right), directly capturing observation-level correlations. This allows sparse models to learn fine-grained relationships more efficiently.

Despite these benefits, sparse input introduces challenges. The high dimensional nature of the sparse input makes it harder for models to learn generalizable representations, and direct exposure to noisy observations makes sparse input models more sensitive to the high noise in HWSN datasets. In contrast, for dense input models, the imputed values that occupy a considerable part of input are obtained by referring to multiple observations. This helps neutralize the noise from individual sensors, making the dense input models more robust to noise.

Based on the above insights, we propose the Context Encoder Spatial Interpolation (CESI) Model with the following contributions:

- CESI is among the first spatial interpolation models tailored for narrow-format sparse input, effectively addressing the heterogeneity in HWSN datasets and achieving significant performance gains.
- We designed a self-supervised context embedding module to handle the sparse input series. This module uses variational inference to learn the probabilistic encoding of the input observations and uses a self-supervised loss signal to achieve an adaptive balance of inductive bias with other modules. Thus, the model's robustness against noise and universality across different tasks is significantly improved.
- We tested our model on three publicly available real-world HWSN datasets from different fields and with different characteristics. Compared to the baselines, whose performance is shaky across different datasets, our model consistently outperforms baselines on all three datasets.

6.4.1 Preliminaries

6.4.1.1 Notations. We regard a HWSN dataset $D = \{F_j \mid j = 1,2,...,n\}$ as a collection of Frames F_j. Each Frame $F_j = \{O_i \mid i = 1,2,...,m\}$ contains all the Observations O_i recorded at a same time, which an example is illustrated as the table in Figure 6.6 (right). Each Observation $O_i = (P,C,V)$ is a triplet of a one-hot encoded Property P, a two- or three-dimensional Coordinate C, and a Value V, which an example is illustrated as a row in Figure 6.6 (right).

We refer to the Property that needs to be interpolated as the Target Property, abbreviated as P_{tgt}. For our model, we only consider one Target Property at a time. Since the spatial distribution of the Target Property is usually not only affected by spatial correlation but also correlated with some other Properties, HWSN datasets also observe these correlated Properties. They are called Support Properties, abbreviated as P_{sup}. Thus, a Frame F can be further divided into two parts: Target Sequence F_{tgt} includes all the Observations of P_{tgt} and Support Sequence F_{sup} includes all the Observations of P_{sup}.

6.4.1.2 Spatial Interpolation Task. Given an input Frame F' (F' may not in D), the spatial interpolation task is to predict the value V' of the P_{tgt} at any arbitrary target location C'. The basis for interpolation comes from the spatial correlation with the known values in $F^{'}_{tgt}$ and the effect of $F^{'}_{sup}$ on this correlation, which can be learned from Frames provided in D.

6.4.2 Framework

The main challenge of sparse input models on HWSN datasets is the contradiction between the requirement of adaptively discovering complex correlations and defective datasets, mainly manifested in low spatial coverage limited by the amount of the sensor and high noise due to the introduction of low-cost sensors. Such problems are usually solved in other fields by obtaining more data sources or using data augmentation approaches. However, in spatial interpolation tasks, such methods are generally limited. We can no longer return to the past to collect data from more locations, and we also lack prior knowledge of those Target Properties affected by complex systems for artificially creating more data. It's worth noting that some heuristics widely used in other fields, such as translation and transposition, are also risky in fields like meteorology, where spatial correlations are significantly affected by longitude, latitude, and azimuth.

These challenges necessitate a robust model design. Models with weak inductive biases, like Transformers [121], excel at capturing complex correlations but heavily depend on data quality and quantity, making their results unstable on HWSN datasets. Conversely, models with strong inductive biases, such as KCN [3] or even Inverse Distance Weighting Interpolation, while based on simple assumptions, perform surprisingly strongly on specific datasets. Nev-

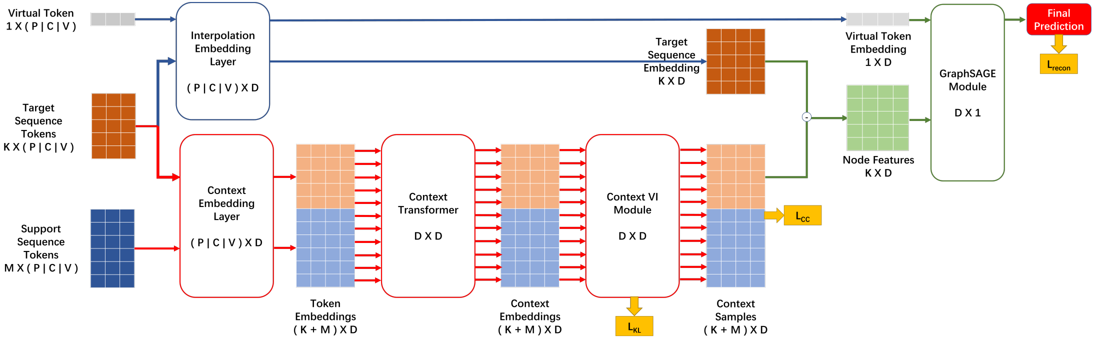


Figure 6.7: An overview of the CESI model. The left half is the GI Module, which mainly models the spatial correlations. The right half shows the TCE Module, which models the influence of the Support Sequence on the spatial correlations.

ertheless, they also risk their inductive biases being mismatched with the dataset. To address this, we propose a hybrid strategy: a strong inductive bias module serves as the backbone, complemented by a weak inductive bias module as an auxiliary component. A self-supervised signal dynamically balances the two modules, enabling better adaptation to different tasks.

6.4.2.1 Transformer-based Context Embedding (TCE) Module. We design a Transformer-based module as our auxiliary component, whose structure is illustrated as the right part of Figure 6.7. It learns the observation-level influence of the Support Properties on the spatial correlation of the Target Property. This influence is eventually encoded as Context Samples, which are subsequently used to correct the inputs of the GraphSAGE Module.

The TCE Module starts with input centering, that is, replacing the absolute coordinates in each observation of the input Frame F with its relative coordinates to the target location C': $F_{cen} = \{(P_i, C_i - C', V_i) \mid i = 1, 2, ..., m\}$. With input centering, we hide the information of specific coordinates in the input Frame, forcing the TCE Module to concentrate on more generalizable spatial correlations. F_{cen} is then embedded by a multi-layer perceptron (MLP) and further processed by the Context Transformer. The Context Transformer is without positional embedding, making it order-independent for the input sequence. With the multi-head self-attention mechanism, each output token of the Context Transformer is obtained after referring to the information of all tokens in F_{cen}. In our design, the underlying intuition here is: for each input token, assuming that all other tokens are noise-free, how much should we adjust its embedding?

The output of the Context Transformer is a deterministic encoding that maps each token to a specific point in the latent space. As the reconstruction error decreases, the model risks overfitting noise in the dataset, leading to degraded performance. We use Variational Inference (VI) to learn a smooth, probabilistic latent space to address this. In probabilistic encoding, the data with noise is treated as a sample of the learned distribution. We construct a continuous and smooth latent space by repeatedly sampling from the learned distribution and ensuring these samples yield consistent outputs. This approach significantly enhances the model's generalization ability while providing meaningful uncertainty estimates for the final output. Specifically, we assume the posterior distribution in the latent space $q(z|x)$ follows a Gaussian distribution. The deterministic encoding x is passed through two MLPs to predict the mean μ and variance σ^2 of $q(z|x)$, respectively. Using the reparameterization trick, we sample a random Context Sample $z = \mu + \sigma\varepsilon$ from $q(z|x)$, where $\varepsilon \sim \mathcal{N}(0, \mathbf{I})$. To align the learned posterior $q(z|x)$ with the standard normal prior $p(z) \sim \mathcal{N}(0, \mathbf{I})$, we minimize their KL divergence: $L_{KL} = D_{KL}(q(z|x) \,||\, p(z))$.

6.4.2.2 GraphSAGE-based Interpolation (GI) Module. We select GraphSAGE as our backbone module, whose structure is illustrated as the left part of Figure 6.7. GraphSAGE assumes that the message-passing process follows the graph's topology, exchanging information within local neighborhoods through shared aggregation and update functions. This represents a relatively strong inductive bias. First, we construct a Virtual Token representing the target location in the format of an observation, in which the Value is filled as zero: $O_v = (P_{tgt}, C', 0)$. The Virtual Token, along with the tokens in F_{tgt}, is then encoded by an MLP, which is the Interpolation Embedding Layer in Figure 6.7, resulting in the Virtual Token Embedding and the Target Sequence Embeddings. Next, we use the Context Samples from the TCE Module to correct their corresponding Target Sequence Embeddings, resulting in Node Features. The Virtual Token Embedding and the Node Features are then together treated as the node feature matrix of the input graph. The adjacency matrix of the input graph is constructed using the k-nearest neighbors heuristic. Then, GraphSAGE is applied to process this graph. Finally, the GraphSAGE output corresponding to the Virtual Token is fed into an MLP Head to produce the interpolation result V'. We use the mean absolute error between V' and the label L as part of the supervisory signal for model training, named reconstruction loss: $L_{recon} = MAE(V', L)$.

6.4.2.3 Context Correction Loss. In addition to L_{recon} and L_{KL}, we introduce another self-supervision loss signal, named Context Correction Loss L_{CC}, to automatically balance the inductive bias of the two modules. It is the average of the L1 Norm of all Context Samplings: $L_{CC} = \frac{1}{n}\sum_{i=1}^{n} \|CS_i\|_1$.

Introducing L_{CC} can bring the following benefits that stabilize the model's performance. First, since the input of the GraphSAGE Module is a linear combination of Target Sequence Embeddings and Context Samples, by limiting the Context Samples to the global minimum, the L_{CC} can make sure that the GraphSAGE Module dominates the training when backpropagating the L_{recon}. This ensures that the GraphSAGE module keeps being the central component of the pipeline. Second, since the Context Samples are sampled from a Gaussian distribution q learned by the Context VI Module, minimizing L_{CC} can constrain the standard deviation of q, preventing the model from identifying the major part of the input as noise and converging to suboptimal results. Third, the L_{CC} encourages the TCE Module to correct the inputs with the minimum possible corrections. This can be thought of as an Occam's razor-based heuristic. When a simple and a complex correction achieves similar results on a poorly sampled dataset, we will prefer the simpler one, thus reducing the overfitting.

The final loss Signal of the model pipeline is a linear combination of L_{recon}, L_{KL}, and L_{CC}: $L = L_{recon} + L_{KL} + L_{CC}$.

Table 6.7: Comparison of Datasets Included in this Study

Name	Sensor Type	Noise Level	P_{sup} Channels	Average F Length	Spatial Coverage Rate[1]	Missing rate[2]
SAQN	All fixed-location	High	8	200.31	12.30%	52.54%
ABO	All movable	Low	2	460.52	4.68%	0.38%
Marine	mixed	Pass quality inspection	5	339.62	97.96%	21.15%

1. How many grids have been observed at least once in the entire dataset
2. How many input cells are missing when expressed as dense input

6.4.3 Experiment

6.4.3.1 Experimental Setup.

Datasets. We evaluate CESI on three publicly available real-world datasets: the SmartAQnet dataset (SAQN) [63], the NOAA Aircraft Based Observation dataset (ABO) [86], and the Copernicus In-situ Marine Observation dataset (Marine) [105]. Table 6.7 provides detailed dataset information.

The SAQN dataset is a typical fixed-location HWSN dataset that monitors urban air quality and meteorological conditions. It is characterized by a high missing rate and considerable noise due to deploying numerous low-cost sensors. Furthermore, its spatial coverage is limited as it relies exclusively on fixed-location sensors. In contrast, the ABO and Marine datasets reflect the trend of incorporating movable sensors in HWSN datasets for higher spatial coverage, which leads to more complex sensor topologies. The ABO dataset, which monitors meteorological parameters using sensors mounted on commercial aircraft, is distinguished by its low noise and extremely low missing rate. However, despite its larger number of observations in each Frame, the ABO dataset still exhibits limited spatial coverage as it is the only three-dimensional dataset included in our analysis. The Marine dataset, on the other hand, measures hydrological and meteorological parameters. Its use of a wide array of movable sensors results in high spatial coverage. This dataset also resembles a traditional dataset, given its relatively low missing data rate and the implementation of strict quality inspection processes. Experiments using the Marine dataset also provide an opportunity to evaluate the effectiveness of our model on more conventional datasets.

Baselines. We involve GraphSAGE [41] and Transformer [121] into baselines, as they are the base components of our model. From the GNN-based spatial interpolation models, we involve GAT [122], KSAGE [3], PE-SAGE [58], LSPE [33], and SPONGE [85]. From the attention-based spatial interpolation models, we involve SSIN [65], and SMACNP [6]. We will publish all the experiment codes if the paper is accepted.

6.4.3.2 Overall Performance. After random searches on hyperparameters, each model was evaluated with four-fold leave-one-area-out cross-validations and five random seeds (1, 2, 3, 4, and 5). Table 6.8 shows the overall performance. The evaluation metrics are Mean Absolute Error (MAE) and R^2, commonly used in spatial interpolation tasks. All results are summarized following the evaluation protocol described in section 2.3.3.

Q1: Which model demonstrates the best overall performance?

A1: CESI achieves the best average MAE and R^2 on all datasets. On the

Table 6.8: Overall Result of all models. Bold indicates the best performer, underline indicates the second place

Model	ABO		SAQN		Marine	
Metrics	MAE(°C)	R^2	MAE($\mu g/m^3$)	R^2	MAE(°C)	R^2
GraphSAGE	10.293 ± 0.044	0.451 ± 0.004	5.863 ± 0.048	0.317 ± 0.009	1.993 ± 0.038	0.631 ± 0.012
Transformer	1.811 ± 0.639	0.972 ± 0.025	6.041 ± 0.437	0.184 ± 0.104	0.971 ± 0.144	0.903 ± 0.027
KSAGE	14.268 ± 0.021	0.012 ± 0.002	5.535 ± 0.048	0.301 ± 0.010	3.128 ± 0.018	0.198 ± 0.007
PE-SAGE	3.302 ± 0.258	0.927 ± 0.008	6.115 ± 0.243	0.217 ± 0.047	1.315 ± 0.042	0.835 ± 0.011
LSPE	13.844 ± 0.424	-0.390 ± 0.409	6.205 ± 0.115	0.171 ± 0.030	1.660 ± 0.084	0.721 ± 0.026
SPONGE	3.918 ± 0.296	0.913 ± 0.013	6.388 ± 0.138	0.249 ± 0.019	1.593 ± 0.071	0.768 ± 0.023
SSIN	18.800 ± 0.469	-0.420 ± 0.062	6.197 ± 0.084	0.167 ± 0.034	1.035 ± 0.052	0.893 ± 0.014
SMACNP	3.241 ± 0.281	0.884 ± 0.025	6.237 ± 0.337	0.201 ± 0.062	1.741 ± 0.044	0.287 ± 0.127
CESI	**1.426 ± 0.040**	**0.987 ± 0.001**	**5.362 ± 0.110**	**0.334 ± 0.008**	**0.944 ± 0.036**	**0.910 ± 0.009**

ABO dataset, Transformer and SMACNP rank second and third, respectively, while Transformer and SSIN occupy these positions on the Marine dataset. On the SAQN dataset, however, KSAGE and GraphSAGE take second and third place, as the above models experience significant degradation. In conclusion, CESI consistently outperforms all baselines across all three datasets, highlighting its adaptability.

Q2: Is sparse input a beneficial choice for spatial interpolation?

A2: Sparse input is beneficial but presents challenges. On ABO and Marine datasets, even the Vanilla Transformer surpasses dense input baselines on average performance. However, sparse input models are more sensitive to noise and bias in lower-quality datasets, such as the Transformer failure on the SAQN dataset, and its performance is volatile on all the datasets. CESI effectively addresses this challenge, with MAE standard deviations 93.7%, 74.8%, and 75.0% lower than Transformer on ABO, SAQN, and Marine datasets, respectively. CESI's stability is competitive even against dense input models.

Q3: Why do many models degrade performance on the SAQN dataset?

A3: First, the SAQN dataset only contains fixed-location sensors, coupled with a low spatial coverage rate, resulting in a high location-related bias in the dataset. Models that employ learnable location-based encodings (e.g., PE-SAGE, LSPE, SPONGE, SSIN) are particularly susceptible to these biases, leading to significant performance degradation. Second, the SAQN dataset has the highest heterogeneity and noise level. Models lacking stable inductive bias (Transformer and SMACNP) tend to overfit the noise, resulting in volatile performances. Our model, on the contrary, successfully overcomes these challenges.

Q4: Why is the performance on the ABO dataset so polarized?

A4: Models like GraphSAGE, KSAGE, and SSIN use Euclidean distance-based heuristics for encoding spatial relationships, and unlike PE-SAGE and CESI, they do not incorporate additional location-based embeddings. The hidden inductive bias of such heuristics is the spatial isotropy of the Euclidean distance. However, on the ABO dataset, the Target Property (air temperature) has an evident stratification along the altitude dimension. This reminds us again that we should be cautious when introducing inductive bias into model design.

When the inductive bias of the model is consistent with the actual situation of the dataset, we can learn a good model with less and worse data. However, when the model's inductive bias conflicts with the dataset's actual situation, the model's performance will be negatively affected.

6.4.3.3 Ablation Study. We use the following ablation models to study the effectiveness of each module: CESI w/o L_{KL} model removes the probabilistic encoding and its associated L_{KL}, CESI w/o L_{CC} model removes the Context Correction Loss L_{CC}, and CESI Null model simultaneously removes the both. All experiment settings are the same as above. Table 6.9 shows the results of the ablation study.

On the ABO dataset, both modules contribute to performance improvement. The main contribution comes from probabilistic encoding, while L_{CC} further refines the performance. On the SAQN dataset, the contribution on average MAE from both modules is roughly the same, and L_{CC} provides more stability improvement than probabilistic encoding.

However, our modules had a slight adverse effect on the Marine dataset. The probabilistic encoding and L_{CC} are designed to address bias and noise in datasets. However, these occurred less in the Marine dataset. First, the dataset has undergone strict quality checks, making it generally noise-free. Second, it boasts exceptionally high spatial coverage (up to 97.96%), minimizing location-related bias. This led to misattributions of our modules, where the probabilistic encoding mistakenly interpreted some genuine correlations as noise, resulting in the significant performance drop of CESI w/o L_{CC}. From the performance of CESI and CESI w/o L_{KL}, we observed that L_{CC} effectively served its intended purpose of constraining such misattributions yet did not fully mitigate the performance decline. Nevertheless, as the overall results demonstrated, this did not prevent the model from achieving state-of-the-art performance. This highlights that our model's competitive edge relies not solely on exploiting flawed datasets but also on learning fine-grained observation-level correlations.

We conducted additional experiments on robustness to missing rates and noise using the Marine dataset to validate our explanation.

Table 6.9: Result of Ablation Study. Bold indicates the best performer, underline indicates the second place

Model	ABO		SAQN		Marine	
Metrics	MAE(°C)	R^2	MAE($\mu g/m^3$)	R^2	MAE(°C)	R^2
CESI	**1.426 ± 0.040**	**0.987 ± 0.001**	**5.362 ± 0.110**	**0.334 ± 0.008**	0.944 ± 0.036	0.910 ± 0.009
CESI w/o L_{KL}	2.020 ± 0.187	0.972 ± 0.005	6.018 ± 0.156	0.170 ± 0.030	0.900 ± 0.025	0.915 ± 0.008
CESI w/o L_{CC}	1.489 ± 0.045	0.986 ± 0.001	6.044 ± 0.500	0.214 ± 0.108	0.980 ± 0.019	0.896 ± 0.008
CESI Null	2.285 ± 0.064	0.968 ± 0.002	8.548 ± 1.170	-0.672 ± 0.654	**0.865 ± 0.031**	**0.922 ± 0.005**

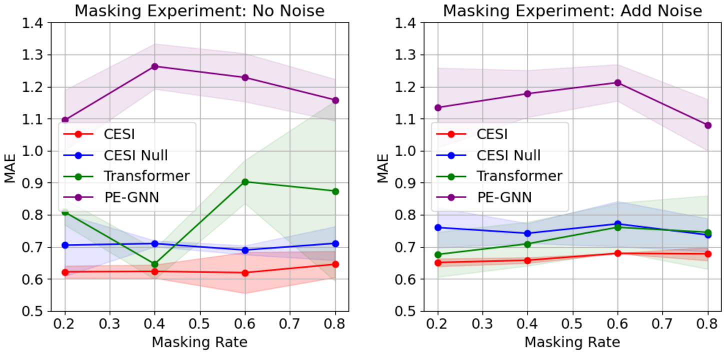


Figure 6.8: Result of Robustness Experiment, the shaded area marks the standard deviation

6.4.3.4 Experiments on Robustness. In the robustness experiment, we randomly mask 20%, 40%, 60%, and 80% of the observations from each Frame in the Marine Dataset to increase its missing rate, and we randomly add multiple Gaussian noise with different standard deviations to varying proportions of data. Then, we train CESI, CESI Null, Transformer, and PE-GNN models on these datasets. We train with random seeds 1, 2, and 3 for each model, respectively. The results are summarized in Table 6.10 and Figure 6.8.

Obviously, (1). the CESI model performs best in all experiments. (2). Although we added noise with different standard deviations to different proportions of data, the noise didn't significantly affect the performance of the CESI model. Since the Gaussian noise added is consistent with the preset of probabilistic encoding, after getting rid of the misattribution, the stability of the model is even improved. (3). Dense input models represented by PE-GNN are hardly affected by the missing rate and noise because the model and data augmentation provide very stable inductive biases. However, as a price, it sacrifices the ability to discover fine-grained correlations, so the overall performance is the worst.

The above concludes that our design works as expected and can maintain the model's performance and stability under different noise and missing rates.

Table 6.10: Result of Robustness Experiment. Bold indicates the best performer, underline indicates the second place

Masking Rate	20%		40%		60%		80%	
Metrics	MAE(°C)	R^2	MAE(°C)	R^2	MAE(°C)	R^2	MAE(°C)	R^2
Without additional noise								
CESI	**0.622 ± 0.019**	**0.874 ± 0.005**	**0.623 ± 0.021**	**0.878 ± 0.006**	**0.619 ± 0.064**	**0.878 ± 0.026**	**0.645 ± 0.041**	**0.872 ± 0.017**
CESI Null	0.705 ± 0.097	0.850 ± 0.040	0.710 ± 0.009	0.843 ± 0.005	0.690 ± 0.014	0.857 ± 0.008	0.711 ± 0.053	0.848 ± 0.019
Transformer	0.809 ± 0.040	0.797 ± 0.013	0.646 ± 0.048	0.864 ± 0.025	0.903 ± 0.067	0.722 ± 0.042	0.874 ± 0.281	0.690 ± 0.234
PE-GNN	1.095 ± 0.094	0.631 ± 0.066	1.263 ± 0.071	0.508 ± 0.054	1.228 ± 0.075	0.525 ± 0.067	1.158 ± 0.065	0.589 ± 0.045
With additional noise								
CESI	**0.651 ± 0.012**	**0.869 ± 0.003**	**0.658 ± 0.009**	**0.867 ± 0.001**	**0.679 ± 0.001**	**0.862 ± 0.001**	**0.678 ± 0.021**	**0.856 ± 0.004**
CESI Null	0.760 ± 0.063	0.834 ± 0.025	0.742 ± 0.031	0.840 ± 0.012	0.771 ± 0.071	0.829 ± 0.028	0.737 ± 0.051	0.841 ± 0.018
Transformer	0.676 ± 0.070	0.859 ± 0.028	0.709 ± 0.069	0.845 ± 0.032	0.761 ± 0.076	0.823 ± 0.033	0.745 ± 0.114	0.826 ± 0.051
PE-GNN	1.134 ± 0.124	0.603 ± 0.089	1.178 ± 0.073	0.594 ± 0.049	1.212 ± 0.057	0.559 ± 0.057	1.080 ± 0.080	0.651 ± 0.047

6.4.4 Summary

We propose the CESI Model for HWSN datasets. Our model directly takes the narrow format sparse input and learns their correlations. Since HWSN datasets usually exhibit small-scale, low spatial sampling rates and considerable noise, we use probabilistic encoding and a self-supervision signal named Context Correction Loss to extract encodings conducive to better generalizing to coordinates not present in the training set. As a result, we effectively improve the model's performance and stability. Experiments across several publicly available real-world HWSN datasets with different characteristics show the CESI Model holds significant potential for broader applications, such as enhancing data-driven decision-making in environmental monitoring, urban planning, and other domains reliant on sparse spatial data.

6.5 Benchmarking Pipeline for Sensor Network Spatial Interpolation Algorithms with the Ground Truth from Remote Sensing Datasets

GNNs have garnered significant attention for their capability to represent and process spatially structured data by modeling sensor networks as graphs. This graph-based perspective enables GNNs to capture local and global dependencies, making them particularly effective for spatial interpolation, especially in unstructured sensor network topologies. However, despite their promise, the application of GNNs in this area remains limited—primarily due to the difficulty in acquiring sufficient ground truth data for training and evaluation. The scarcity of data impedes the development of accurate models and compromises the reliability of model evaluation, creating a significant bottleneck in the broader adoption of advanced spatial interpolation techniques.

Our motivation lies in addressing these challenges and exploring the potential to enhance both the applicability and performance of GNNs in spatial interpolation. To this end, we propose a framework to improve data availability during the evaluation phase, thereby mitigating the limitations imposed by data scarcity. This framework is designed to leverage the strengths of GNNs and maintain compatibility with various model types. Through this work, we aspire to offer a more reliable and scalable solution for spatial interpolation, ultimately advancing capabilities in environmental monitoring and analysis. Our overarching goal is to bridge the gap between state-of-the-art deep learning methods and their practical deployment in sensor networks, ensuring these powerful tools can be effectively applied in real-world contexts.

Despite growing interest in spatial interpolation models, current benchmarking practices for deep learning-based approaches remain insufficient for thorough and reliable performance assessment. Most existing studies rely exclusively on sensor network data, which, while valuable, present several limitations—such as spatial sparsity, uneven sensor placement, and resolution constraints—that can result in incomplete or overly optimistic evaluations of model accuracy. These constraints hinder the development of robust interpolation techniques capable of handling complex geospatial variability.

Additionally, current benchmarking frameworks seldom evaluate model stability across multiple spatial resolutions. Many studies focus on interpolation performance at a single resolution, limiting our understanding of how models generalize across varying levels of spatial granularity. This is particularly problematic for high-resolution applications, such as climate modeling and environmental monitoring, where precise spatial predictions are critical.

Another significant gap is the lack of independent ground truth validation using remote sensing datasets. While sensor networks offer valuable point-based observations, they often fail to capture broader spatial patterns due to their limited coverage and coarse spatial distribution. Interpolation models risk becoming biased toward sensor locations, diminishing their generalizability in practical scenarios.

These limitations underscore the need for an improved benchmarking methodology that integrates high-resolution spatial data and evaluates model performance across multiple scales.

To address these issues, we propose a novel benchmarking approach that extends traditional evaluation practices by incorporating ground truth data from remote sensing sources. Unlike prior studies that rely solely on sensor network measurements, our approach leverages high-resolution remote sensing data to deliver a more comprehensive and dependable evaluation of interpolation models.

Remote sensing offers continuous, large-scale, high-resolution spatial data that significantly enhances the assessment of model performance. We facilitate a more detailed and accurate evaluation across varying spatial scales by comparing model predictions with remote-sensing ground truth. This increases the reliability of model assessments and reduces the risk of performance overestimation due to the limitations of sensor-based data.

Furthermore, our approach allows for comparison of how effectively different models capture spatial patterns. We evaluate several interpolation models and analyze their performance across multiple spatial resolutions. Specifically, model performance is systematically assessed at four additional resolution levels to investigate how spatial granularity impacts predictive accuracy and reliability. This multi-resolution evaluation framework enables us to rigorously examine each model's robustness and scalability.

Ultimately, our proposed benchmarking methodology can establish a new standard for evaluating spatial interpolation models by integrating sensor network data and remote sensing observations. This combined approach enhances model assessment and broadens the applicability of interpolation techniques across diverse geospatial domains.

6.5.1 Pipeline design

Figure 6.9 presents an overview of our proposed benchmarking pipeline framework for sensor network algorithms, incorporating ground truth data from remote sensing datasets. The framework comprises five key components: data preparation, parameter mapping, sampling, GNN model training and testing, and an extended evaluation mechanism.

The first stage involves data cleaning and standardization. Subsequently, the parameters from the remote sensing dataset are aligned with those of the sensor network dataset. A new dataset is then generated that preserves the structural layout of the original sensor network data but replaces its values with corresponding values from the remote sensing dataset. Additional data points from the remote sensing dataset are integrated into the evaluation subset to enhance the evaluation process. These ground truth points are sampled at varying spatial resolutions to enable comprehensive performance assessment across different scales.

6.5.1.1 Dataset. ERA5-Land is a state-of-the-art global reanalysis dataset for land applications, developed under the Copernicus Climate Change Service (C3S) of the European Commission and produced by the European Centre for Medium-Range Weather Forecasts (ECMWF) [84].

ERA5-Land provides a consistent description of the evolution of the water and energy cycles over land throughout its production period. This consistency is achieved through global high-resolution numerical integrations using the ECMWF land surface model. One of the primary advantages of ERA5-Land over ERA5 and the earlier ERA-Interim datasets lies in its enhanced horizontal resolution—9 km globally, compared to 31 km in ERA5 and 80 km in ERA-Interim—while maintaining the exact hourly temporal resolution as ERA5. The improved spatial resolution significantly reduces the global average root mean square error of skin temperature, particularly in coastal regions where fine spatial detail is crucial. With its high spatial and temporal resolution, extended time coverage, and internally consistent variables, ERA5-Land is a valuable resource for hydrological research, initialization of numerical weather prediction and climate models, and various water resource, land, and environmental management applications.

This study uses ERA5-Land as the ground truth data source due to its dense spatial grid and comprehensive set of meteorological and geophysical parameters. The subset employed focuses on the region spanning from the equator to 20°S latitude and from 70°W to 50°W longitude, encompassing various climates and terrains across South America.

6.5.1.2 Data Preparation. The data preparation stage consists of two main steps: data cleaning and standardization of input data from the sensor network and the high-resolution remote sensing dataset.

Factors such as instrument malfunctions, calibration errors, environmental conditions, or data transmission failures can lead to missing values or incomplete records in both datasets. Therefore, data cleaning is essential to appropriately address erroneous, missing, or inconsistent entries. Rather than imput-

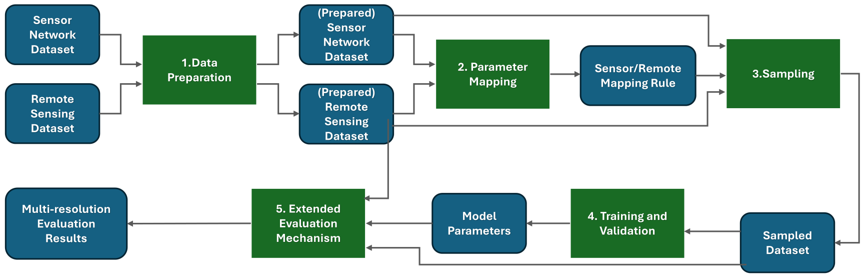

Sensor Network Dataset
Remote Sensing Dataset
1.Data Preparation
(Prepared) Sensor Network Dataset
(Prepared) Remote Sensing Dataset
2. Parameter Mapping
Sensor/Remote Mapping Rule
3.Sampling
Sampled Dataset
4. Training and Validation
Model Parameters
5. Extended Evaluation Mechanism
Multi-resolution Evaluation Results

Figure 6.9: Pipeline design

ing missing values—which may introduce inaccuracies due to such estimates' likely unreliability—we remove incomplete or invalid entries. This ensures that the spatial interpolation is based solely on trustworthy data.

Following data cleaning, all input features are standardized to facilitate model training. Specifically, we apply min-max normalization to each parameter, scaling values to a fixed range, typically [0, 1], based on their respective minimum and maximum values. This normalization process enhances model stability and accelerates convergence during training by ensuring consistent data scaling across inputs.

6.5.1.3 Parameter Mapping. We perform a parameter mapping procedure to align the observed properties in the sensor network dataset with their corresponding counterparts in the remote sensing dataset. Although both datasets provide environmental measurements, they often differ in the specific parameters they record. By establishing this alignment, we can treat remote sensing values as valid proxies for the sensor-based measurements, bridging the gap between the sensor network and remote sensing datasets within our benchmarking framework.

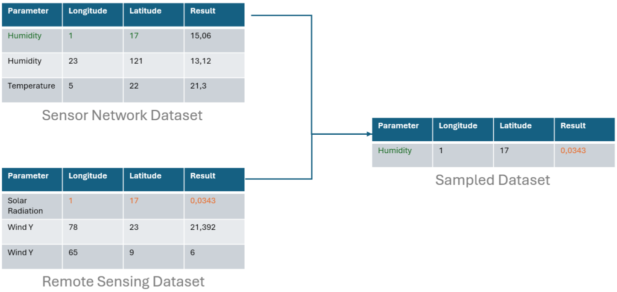
Secondly, this alignment also facilitates a standardized testing framework for comparing different interpolation methods on sensor networks with different topologies, ensuring that all algorithms are evaluated on a standard and comparable basis.

Finally, the mapping process mitigates potential discrepancies arising from differences in measurement techniques, units, or spatial and temporal scales between the two datasets. This step is, therefore, vital to maintaining the integrity, consistency, and comparability of results across all benchmarked models.

6.5.1.4 Sampling and creating new working Dataset. The next step involves creating a new working dataset, referred to as the sampled dataset. This dataset serves as the foundation for training, validating, and evaluating interpolation algorithms, as it integrates the structural characteristics of the sensor network dataset with high-resolution ground truth values obtained from the remote sensing dataset.

The sampled dataset is constructed by preserving the structural layout of the original sensor network dataset—including its spatial and temporal configuration and the organization of its parameters. This approach ensures that the new dataset closely replicates a real-world sensor network's operational constraints and characteristics. However, instead of relying on the original sensor network's recorded values, the sampled dataset replaces them with values extracted directly from the remote sensing dataset.

These remote sensing values are sampled at precisely regularized geographic locations and time intervals corresponding to those represented in the sensor network. This alignment guarantees that the sampled dataset remains spatially and temporally consistent with the original observations while benefiting from the accuracy and completeness of the remote sensing data. The entire sampling process is guided by the parameter mapping procedure outlined in the previous section.



Parameter	Longitude	Latitude	Result
Humidity	1	17	15,06
Humidity	23	121	13,12
Temperature	5	22	21,3

Sensor Network Dataset

Parameter	Longitude	Latitude	Result
Solar Radiation	1	17	0,0343
Wind Y	78	23	21,392
Wind Y	65	9	6

Remote Sensing Dataset

Parameter	Longitude	Latitude	Result
Humidity	1	17	0,0343

Sampled Dataset

Figure 6.10: Process creating Sampling dataset, using Humidity to Solar Radiation mapping rule

6.5.1.5 Training and Validating GNN Models. With the sampled dataset prepared, the next stage of our benchmarking framework involves training and validating GNN-based interpolation models. In this process, the sampled dataset is fed into the GNN models, where each data point is represented as a node, and the spatial relationships between locations are encoded as edges within the graph structure. The node features—such as environmental parameters or sensor readings—serve as input attributes for the model.

6.5.1.6 Extended Evaluation Mechanism. The extended evaluation mechanism represents our proposed benchmarking framework's final and most critical component. Its purpose is to comprehensively assess the performance of interpolation models across varying sampling resolutions, thereby simulating the complexities of real-world deployment. This mechanism enhances traditional evaluation methodologies by incorporating additional ground truth data from the remote sensing dataset, allowing for rigorous model robustness and accuracy testing under diverse spatial scenarios.

Unlike conventional deep learning evaluation practices—which rely exclusively on subsets of the original dataset (portions of the dataset used for training, validation, and evaluation)—our extended evaluation mechanism leverages the full spatial resolution of the remote sensing data. Specifically, it integrates additional ground truth points not included in the sampled dataset used during training. These extra points are strategically selected at different spatial resolutions to construct more challenging and realistic test scenarios. High-resolution sampling allows us to evaluate the model's ability to capture fine-grained spatial patterns. In contrast, low-resolution sampling simulates the conditions of sparse sensor coverage typically found in real-world networks.

Additionally, our evaluation framework adopts a leave-out-one-block validation strategy instead of the more commonly used leave-out-one-sensor method.

Traditional approaches to spatial interpolation—especially in the context of weather prediction—often evaluate model generalization by withholding a single sensor during training and using its data for testing. While this method is straightforward, it has notable drawbacks. Sensor distributions are often uneven, with some areas densely instrumented and others sparsely. Consequently, models trained and tested in such conditions tend to be biased toward regions with high sensor density, failing to generalize effectively to underrepresented or unseen areas.

Moreover, spatial dependencies inherent in geospatial data further undermine the effectiveness of leave-one-sensor-out strategies. Sensors nearby often capture overlapping environmental signals, meaning that even when a single sensor is excluded, nearby sensors may still provide indirect information about the target location. As a result, model performance may appear artificially high, masking actual generalization errors.

David R. Roberts has extensively discussed these issues in the context of ecological and environmental data [97]. He points out that such data typically exhibit temporal, spatial, hierarchical, or phylogenetic dependencies and that ignoring these structures during cross-validation can lead to significant underestimation of prediction error. In particular, non-random cross-validation schemes like block cross-validation are recommended in cases where structured dependence exists, even when residual analyses suggest independence or when models attempt to account for correlation structures through techniques like autoregressive models or mixed effects.

Block cross-validation splits data spatially rather than randomly, providing a more realistic estimate of model performance when extrapolating to new areas or predictor spaces. Roberts' simulations and case studies demonstrate that block cross-validation is consistently more appropriate than random cross-validation for predictive modeling, especially when the goal is to evaluate out-of-distribution generalization or select causal predictors.

Following this guidance, our framework holds out entire spatial blocks instead of individual sensors during evaluation. This approach emphasizes assessing model performance in completely unseen regions, including those with sparse sensor coverage—better reflecting the real-world challenges of deploying interpolation models in data-scarce environments.

While Roberts conducted his simulations using a 50×50 spatial grid, our implementation operates on a more demanding 250×250 grid, with each block comprising a 125×125 region (i.e., one-fourth of the total domain). This higher-resolution setting allows for a more rigorous examination of model robustness and scalability in large-scale geospatial applications.

The workflow is as follows:

1. Train the model using data from three holdouts (blocks).

2. Evaluate the model on the remaining holdout.

3. Repeat the process until each holdout has been used as the evaluation set

once.

This strategy ensures that the evaluation is not biased by the specific characteristics of any single holdout and provides robust performance metrics.

6.5.2 Experiments

Section 6.5.2.1 presents the results of 600 experiments conducted in this research. Section 6.5.2.2 will present a detailed analysis and discussion of the results.

6.5.2.1 Results. The evaluation results presented in Table 6.11 summarize the performance of various models across multiple spatial resolutions, ranging from the original sensor configuration to grid sizes of 16x16, 32x32, 64x64, and 125x125. Each model's performance is evaluated based on three key metrics: Mean Absolute Error (MAE), R-squared (R2), and Mean Absolute Percentage Error (MAPE). These metrics provide a comprehensive view of models' accuracy, goodness of fit, and prediction error.

To ensure the results' robustness, the evaluation is carried out across four different holdouts and five distinct random seeds for each spatial resolution configuration. This approach accounts for variations in model performance due to different data splits and random initializations.

6.5.2.2 Analysis.

Models Performance Analysis.

1. **How does the performance of different models vary between fine-grained and coarse-grained resolutions?** Model performance varies depending on the resolution, with no single model consistently outperforming others across all cases. However, PEGSAGE demonstrates strong overall performance, maintaining good results even at the lowest resolution in the regular grid (16×16). In contrast, GAT excels at the sparsest resolutions (original resolutions) but experiences significant performance drops at coarser scales. While GAT recovers somewhat at finer

Table 6.11: Evaluation results of the models at different spatial resolutions: Original, 16x16, 32x32, 64x64, and 125x125.

Model	**Original**			**16x16**			**32x32**			**64x64**			**125x125**		
	MAE($\mu g/m^3$)	R2	MAPE	MAE($\mu g/m^3$)	R2	MAPE	MAE($\mu g/m^3$)	R2	MAPE	MAE($\mu g/m^3$)	R2	MAPE	MAE($\mu g/m^3$)	R2	MAPE
GCN	1.2175	0.761	0.05	2.573	0.415	1.0968	2.5805	0.424	0.872	2.545	0.368	0.868	2.545	0.365	0.882
GSAGE	1.2525	0.7625	0.0495	2.571	0.465	1.016	2.519	0.4625	0.8435	2.5665	0.4585	0.864	2.537	0.4625	0.671
KSAGE	1.308	0.7365	0.051	2.6025	0.427	1.068	2.5965	0.4405	0.849	2.3805	0.3955	0.7945	2.357	0.3975	0.7895
PEGSAGE	1.123	0.7995	0.045	2.447	0.4	1.0725	2.545	0.41	0.6545	2.5425	0.4085	0.6705	2.5425	0.4085	0.6705
GAT	1.0935	0.782	0.058	2.78	0.3855	1.136	2.6455	0.446	0.877	2.614	0.427	0.887	2.655	0.447	0.881
NODEFORMER	1.3085	0.718	0.0515	2.7245	0.3865	1.082	2.7335	0.37	0.8645	2.7005	0.3555	0.8885	2.671	0.4095	0.8555

resolutions, this fluctuation prevents it from ranking consistently well across all settings. Therefore, in terms of overall stability, PEGSAGE is more robust across different resolutions, while GAT is highly effective at preserving fine details but struggles at lower resolutions.

2. **How do models compare regarding robustness across different resolutions?** While no single model consistently outperforms others across all resolutions, some models demonstrate greater adaptability to changes in resolution. This variability highlights the importance of benchmarking, as performance depends significantly on the resolution used. The results suggest that specific models are more robust to resolution changes, whereas others excel only under particular conditions. Generally speaking, PEGSAGE was the best at adapting to the different resolutions.

3. **How can we explain GAT's performance, particularly the significant decrease when transitioning from the original resolution to 16x16, followed by gradual stabilization as the resolution continues to decrease?** GAT operates on graph-structured data, leveraging masked self-attentional layers to address the shortcomings of prior methods (mainly GCN) based on graph convolutions or their approximations. By stacking layers in which nodes can attend over their neighborhoods' features, GAT can (implicitly) specify different weights to different nodes in a neighborhood without requiring costly matrix operation (such as inversion) or depending on knowing the graph structure upfront. GAT's significant performance drop when transitioning from the original resolution to 16x16 can be attributed to its attention mechanism's reliance on node representations, where the neighborhood structure becomes increasingly sparse and less informative as resolution decreases. This may lead to less meaningful attention, especially when the neighborhood size is large.

4. **Why does PEGSAGE perform well on MAPE and MAE while showing relatively weaker results on R2, and why is PEGSAGE very stable from resolution to resolution?** PEGSAGE excels in capturing local spatial relationships, leading to substantial performance on MAPE and

MAE. One of the reasons may be that PEGSAGE explicitly incorporates spatial context and correlation into the models. Also, PEGSAGE was built on recent advances in geospatial auxiliary task learning and semantic spatial embeddings - PEGSAGE learns a context-aware vector encoding of the geographic coordinates and predicts spatial autocorrelation in the data in parallel with the main task. [58] Also, the creators of PEGSAGE explicitly show the effectiveness of their approach on spatial interpolation tasks, which is also what we see in our results. That can also be a reason for a very stable performance across all resolutions. However, relatively lower R2 in compassing with other models may suggest that the model has difficulties capturing global variance, possibly due to its focus on local dependencies.

5. **What performance shows GCN, the most popular GNN model?** GCN is the most cited paper in the GNN literature and the most commonly used architecture in real-life applications [54]. GCN delivered average performance on MAE and MAPE and struggled with R2 in our experiments, likely due to its limited capacity to capture long-range dependencies and not directly supporting edge features. While to this day, GCN remains a widely used model. Its popularity has recently declined as newer architectures address limitations, such as over-smoothing and scalability issues. The decreasing proportion of papers on GCN reflects a shift towards more advanced models with different ways of overcoming GCN limitations [130]. However, GCN can still be a fundamental baseline for measuring progress in graph-based learning.

6. **Why did NODEFORMER exhibit poor performance in our experiments, and what factors related to its architecture and design could explain these results?** Although it is impossible to give one specific answer, we propose several potential issues and explanations:

   - As stated in the paper [131], one of NODEFORMER's main innovations is its ability to learn latent graphs and perform message passing on those, which might simplify the graph structure too much for specific tasks, such as for relatively more minor datasets like the SAQN sensor network dataset.

- NODEFORMER is designed to scale efficiently to large graphs with linear complexity. However, this scalability could come at the cost of some precision. The model's use of latent graphs and the custom kernelized Gumbel-Softmax operator could lead to suboptimal performance where finer details or more specific connections between nodes are essential.

- *All-Pair Message Passing on Layer-Specific Adaptive Structures* enables NODEFORMER to propagate features across a latent graph that can potentially connect all nodes. This differs from the local propagation approach in traditional GNNs, which aggregate embeddings only from neighboring nodes. However, for spatial interpolation tasks, embedding from neighboring nodes makes more sense and, based on our experiments, results in a better performance.

Pipeline Analysis.

1. **Why is it important to evaluate models across multiple resolutions rather than relying on original resolution results?** The problem with considering only the original resolution is that it can not be enough to assess the model's performance properly. Our pipeline enables a more comprehensive evaluation by systematically comparing model performance at multiple resolutions using the additional ground truths the remote sensing dataset provides.

2. **What are the limitations of traditional evaluation methods?** Traditional methods struggle to handle complex and dynamically changing spatial relationships, making them less effective in highly heterogeneous regions. They rely on simple prior assumptions, which may not always hold in real-world scenarios. For example, traditional benchmarks do not account for variations in the spatial distribution of sensors, making them prone to overestimating a model's effectiveness.

3. **How does our pipeline address these shortcomings?** Our benchmarking pipeline introduces a multi-resolution evaluation approach, assessing models across varying spatial granularities. This ensures that performance evaluations reflect interpolation accuracy and model robustness across scales. Additionally, the pipeline integrates real-world sensor placement, making the assessment more representative of practical applications.

4. **What role does remote sensing data play in our evaluation pipeline?** Remote sensing data provides a rich ground truth reference that enables us to evaluate interpolation quality at high spatial resolutions. This is particularly useful for identifying cases where traditional sparse sensor-based evaluations might overlook interpolation biases. Our pipeline ensures a more reliable and scalable benchmarking approach by incorporating additional ground truth from remote sensing data.

5. **What does the variability in model rankings tell us about the relationship between model architecture and spatial resolution?** The variability in model rankings demonstrates that the effectiveness of a model is highly dependent on the spatial resolution of the dataset, highlighting the intricate relationship between model architecture and the scale of spatial data. Different models prioritize correlations at various spatial scales. For example, GAT performed exceptionally well at the original resolution but showed mixed results at coarse resolutions, suggesting that its attention mechanism is more suited to datasets with moderate spatial complexity. This indicates that models that can better capture spatial relationships at varying scales may be more robust across different resolutions.

6. **Did the models maintain their performance trends when transitioning between resolutions? For example, did specific models consistently improve as the resolution increased, while others struggled or showed inconsistent behavior across different resolutions?** Usually, the models either continued their performance trends when transitioning between resolutions or plateaued at a certain point, meaning that go-

ing to a finer resolution usually resulted in little performance difference. However, there were some exceptions, with some models exhibiting performance jumps moving to a specific resolution.

For example:

- KSAGE MAE performance jumped from 32×32 to 64×64 resolution and remained roughly the same when transitioning to an even finer 125×125 resolution.
- GCN R2 performance jumped from 32×32 to 64×64 resolution and then improved slightly when moving to the finer 125×125 resolution.
- NODEFORMER R2 performance consistently decreased when moving to finer resolutions from 16×16 to 32×32 and then from 32×32 to 64×64. However, performance suddenly improved by a large margin when transitioning from 64×64 to 125×125, making NODEFORMER's performance at this resolution the best, apart from its original resolution.
- GSAGE MAPE performance jumped when going from 64×64 to 125×125, even though it had previously decreased when transitioning from 32×32 to 64×64.

These sudden performance jumps occurred across different models and error metrics. However, we observe a clear trend where such jumps often happen at the finest resolutions (64×64 or 125×125). The exception is MAPE, where all models showed a significant performance jump from 16×16 to 32×32. Additionally, it is essential to note that these performance jumps can occur even when the overall trend suggests that a model's performance should decrease at finer resolutions.

7. **What is the best and worst resolution for evaluating spatial interpolation models?** Our pipeline is designed to assess the spatial frequencies that a model prefers when modeling the original field using different ground truth resolutions. Therefore, it is not about identifying a "best" or "worst" resolution. Instead, multiple resolutions are evaluated to assess how well the models adapt to varying levels of spatial detail. Each

model may prefer specific resolutions over others, depending on how well it can capture the underlying spatial patterns at different scales. The key is to evaluate model performance across multiple resolutions, as no single resolution is inherently the best or worst for all models.

6.5.3 Summary

Using our proposed benchmarking framework, this paper benchmarked six different GNN models on a spatial interpolation task. We uncovered key insights into the capabilities and limitations of commonly used GNN architectures through a systematic analysis of model performance across varying spatial granularities.

One of the most significant findings is that no single model consistently outperforms others across all spatial resolutions. This performance variability highlights the critical role of spatial resolution in model selection, which is often overlooked in conventional evaluation settings. Selecting the "best" model based solely on its performance at a single resolution—typically the original sensor topology—can lead to suboptimal decisions. This finding underscores the value of multi-resolution benchmarks like the one we propose, which expose each model's nuanced strengths and weaknesses. Adopting a comprehensive evaluation approach provides a more informed and practical foundation for selecting the most suitable model for specific real-world scenarios.

6.6 Chapter Summary

In this chapter, we presented a series of spatial analysis models designed to address the key challenges of hybrid sensor network datasets in urban aerosol distribution prediction. Unlike conventional spatial interpolation methods, which often struggle with heterogeneous sensor distributions, missing data, and feature imbalance, our models incorporate adaptive learning strategies to enhance robustness and generalization.

We introduced three specialized models, each targeting a distinct challenge. The Context Encoder Spatial Interpolation (CESI) model effectively extracts meaningful spatial correlations from sparse and unevenly distributed sensor data, ensuring that predictions remain reliable even in areas with limited observations. The Information Segmentation Spatial Interpolation (ISSI) model employs self-supervised learning to mitigate the effects of latent unobserved factors, reducing uncertainty caused by missing environmental influences. Lastly, the Feature Deviation Embedding Graph Spatial Interpolation (FDE-GSI) model tackles feature imbalance by dynamically adjusting graph-based representations, ensuring that underrepresented sensor patterns do not degrade model performance.

Through extensive evaluations of real-world datasets, we demonstrated that our proposed models significantly improve prediction accuracy, robustness, and stability compared to traditional spatial interpolation techniques. More importantly, our results highlight the necessity of tailored spatial analysis methods in hybrid sensor networks, where sensor heterogeneity and missing data can severely impact prediction quality if not adequately addressed.

7 Conclusion

Urban aerosol distribution modeling is crucial in air quality monitoring, environmental policymaking, and urban planning. However, existing spatiotemporal analysis models face substantial challenges due to the inherent limitations of hybrid sensor network datasets. These datasets suffer from heterogeneous sensor deployments, measurement uncertainties, and imbalanced feature distributions, hindering the generalization and reliability of data-driven approaches.

This dissertation addresses these challenges by introducing the following contributions: This dissertation addresses these challenges by introducing the following contributions:

Contribution 1: CFD-Based Data Augmentation for Urban Aerosol Prediction. A key limitation in urban aerosol modeling is insufficient, high-quality training data. Many traditional data augmentation methods used in other domains, such as image transformations, are inapplicable to geospatial data due to its sensitivity to absolute coordinates, orientation, and environmental conditions. This dissertation explores the potential of CFD-based simulations to generate realistic synthetic data for aerosol modeling.

To overcome the high dependency of CFD models on precise input data, we designed a simplified CFD framework that relies on easily accessible meteorological, urban morphology, and traffic emission data. Our experiments show that, despite some discrepancies, the generated synthetic data closely approximates real-world aerosol distributions. We further developed a GNN-based CFD surrogate model to make them more feasible for large-scale data augmentation. This model accelerates CFD-based data generation by learning to approximate CFD outputs while preserving key spatial relationships in pollutant dispersion. The graph structure learning approach enables adaptive graph adjustments, reducing error accumulation in multi-step inference.

Contribution 2: Spatiotemporal Analysis for Hybrid Sensor Network Datasets. While low-cost sensors enhance the spatial coverage of hybrid sensor networks, they also introduce noise, drift, and inconsistencies, raising concerns about their overall impact on predictive modeling. However, our findings reveal that low-cost sensors significantly enhance predictive performance when adequately handled, reinforcing their value despite their inherent limitations.

To tackle the challenges posed by hybrid sensor network datasets and to improve the robustness and generalization of spatiotemporal analysis models, we developed a suite of advanced spatiotemporal modeling techniques:

- Neural Kernel Network Deep Kernel Learning (NKNDKL) model: This model Leverages Gaussian Process Regression (GPR) to capture long-range temporal dependencies and quantify uncertainty. It utilizes the NKN kernel to overcome the challenge of designing appropriate kernel functions without prior knowledge of the complex underlying system.

- Information Segmentation Spatial Interpolation (ISSI) model: Employs self-supervised learning to mitigate uncertainties arising from unobserved Latent Context Information within sensor networks.

- Feature Deviation Embedding Graph Spatial Interpolation (FDE-GSI) model: Addresses feature imbalance by implementing Feature Deviation Embedding alongside an adaptive information bottleneck mechanism, ensuring the model effectively learns from underrepresented patterns.

- Context Encoder Spatial Interpolation (CESI) model: Effectively mines sparse and heterogeneous sensor data, ensuring robust interpolation across irregularly distributed observations.

We evaluate our proposed models on multiple real-world datasets, demonstrating consistent accuracy, robustness, and adaptability. Unlike existing models that exhibit significant performance fluctuations across datasets, our approaches achieve state-of-the-art performance with better stability and generalization. These results highlight the effectiveness of our design choices and their potential for real-world deployment in urban aerosol monitoring applications.

Conclusion. In summary, this dissertation presents a comprehensive approach to tackling hybrid sensor network data limitations that hinder urban aerosol prediction systems. We provide solutions that enhance predictive accuracy, robustness, and computational efficiency by leveraging CFD-based data augmentation and robust spatiotemporal modeling strategies.

Future research can further explore integrating multimodal data sources (e.g., satellite imagery and meteorological simulations) to enhance prediction accuracy, developing adaptive sensor network calibration techniques to improve low-cost sensor reliability, and exploring federated learning to enable cross-city aerosol prediction while maintaining data privacy. By addressing these challenges, we are moving closer to scalable, real-time urban aerosol monitoring systems, ultimately contributing to more informed environmental policies and healthier urban living conditions.

Bibliography

[1] A. Akhatova, A. Kassymov, M. Kazmaganbetova, and L. Rojas-Solorzano. Cfd simulation of the dispersion of exhaust gases in a traffic-loaded street of astana, kazakhstan. *Journal of Urban and Environmental Engineering*, 9(2):158–167, 2015.

[2] J. L. Allen, G. Oberdorster, K. Morris-Schaffer, C. Wong, C. Klocke, M. Sobolewski, K. Conrad, M. Mayer-Proschel, and D. Cory-Slechta. Developmental neurotoxicity of inhaled ambient ultrafine particle air pollution: Parallels with neuropathological and behavioral features of autism and other neurodevelopmental disorders. *Neurotoxicology*, 59: 140–154, 2017.

[3] G. Appleby, L. Liu, and L.-P. Liu. Kriging convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3187–3194, 2020.

[4] N. Ashgriz and J. Mostaghimi. An introduction to computational fluid dynamics. *Fluid flow handbook*, 1:1–49, 2002.

[5] T. Bai and P. Tahmasebi. Graph neural network for groundwater level forecasting. *Journal of Hydrology*, 616:128792, 2023.

[6] L.-L. Bao, J.-S. Zhang, and C.-X. Zhang. Spatial multi-attention conditional neural processes. *Neural Networks*, 173:106201, 2024.

[7] L. Belkhiri, A. Tiri, and L. Mouni. Spatial distribution of the groundwater quality using kriging and co-kriging interpolations. *Groundwater for Sustainable Development*, 11:100473, 2020.

[8] B. Blocken. Computational fluid dynamics for urban physics: Importance, scales, possibilities, limitations and ten tips and tricks towards accurate and reliable simulations. *Building and Environment*, 91:219–245, 2015.

[9] J. Brandstetter, D. Worrall, and M. Welling. Message passing neural pde solvers. *arXiv preprint arXiv:2202.03376*, 2022.

[10] S. Brody, U. Alon, and E. Yahav. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021.

[11] A. Bryutkin, J. Huang, Z. Deng, G. Yang, C.-B. Schönlieb, and A. Aviles-Rivero. Hamlet: Graph transformer neural operator for partial differential equations. *arXiv preprint arXiv:2402.03541*, 2024.

[12] M. Budde, R. El Masri, T. Riedel, and M. Beigl. Enabling low-cost particulate matter measurement for participatory sensing scenarios. In *Proceedings of the 12th international conference on mobile and ubiquitous multimedia*, pages 1–10, 2013.

[13] M. Budde, L. Zhang, and M. Beigl. Distributed, low-cost particulate matter sensing: scenarios, challenges, approaches. In *Proceedings of the 1st International Conference on Atmospheric Dust*, pages 230–236, 2014.

[14] M. Budde, M. Köpke, and M. Beigl. Robust in-situ data reconstruction from poisson noise for low-cost, mobile, non-expert environmental sensing. In *Proceedings of the 2015 ACM International Symposium on Wearable Computers*, pages 179–182, 2015.

[15] M. Budde, T. Riedel, M. Beigl, K. Schäfer, S. Emeis, J. Cyrys, J. Schnelle-Kreis, A. Philipp, V. Ziegler, H. Grimm, et al. Smartaqnet: remote and in-situ sensing of urban air quality. In *Remote Sensing of Clouds and the Atmosphere XXII*, volume 10424, pages 19–26. SPIE, 2017.

[16] M. Budde, A. Schankin, J. Hoffmann, M. Danz, T. Riedel, and M. Beigl. Participatory sensing or participatory nonsense? mitigating the effect of human error on data quality in citizen science. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3):1–23, 2017.

[17] M. Budde, A. D. Schwarz, T. Müller, B. Laquai, N. Streibl, G. Schindler, M. Köpke, T. Riedel, A. Dittler, M. Beigl, et al. Potential and limitations of the low-cost sds011 particle sensor for monitoring urban air quality. *ProScience*, 5(6):12, 2018.

[18] R. Burnett, H. Chen, M. Szyszkowicz, N. Fann, B. Hubbell, C. A. Pope III, J. S. Apte, M. Brauer, A. Cohen, S. Weichenthal, et al. Global estimates of mortality associated with long-term exposure to outdoor fine particulate matter. *Proceedings of the National Academy of Sciences*, 115(38):9592–9597, 2018.

[19] Y. Cao, M. Chai, M. Li, and C. Jiang. Efficient learning of mesh-based physical simulation with bsms-gnn. *arXiv preprint arXiv:2210.02573*, 2022.

[20] F. Carotenuto, A. Bisignano, L. Brilli, G. Gualtieri, and L. Giovannini. Low-cost air quality monitoring networks for long-term field campaigns: A review. *Meteorological Applications*, 30(6):e2161, 2023.

[21] X. Chen, M. Wang, Z. Jiang, Y. Zhang, L. Zhou, J. Liu, H. Liao, H. Worden, T. He, D. Jones, et al. Large discrepancy between observations and simulations: Implications for urban air quality in china. *arXiv preprint arXiv:2208.11831*, 2022.

[22] W. Cheng, Y. Shen, Y. Zhu, and L. Huang. A neural attention model for urban air quality inference: Learning the weights of monitoring stations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[23] D. Cho, C. Yoo, J. Im, Y. Lee, and J. Lee. Improvement of spatial interpolation accuracy of daily maximum air temperature in urban areas

using a stacking ensemble technique. *GIScience & Remote Sensing*, 57 (5):633–649, 2020.

[24] A. Cini, I. Marisca, and C. Alippi. Filling the g_ap_s: Multivariate time series imputation by graph neural networks. *arXiv preprint arXiv:2108.00298*, 2021.

[25] T. Cleary, J. Yang, and M. Fernandez. New research on diffusion of carbon monoxide through gypsum wallboard. *Gaithersburg, MD*, 2014.

[26] T. Cui, D. Pagendam, and M. Gilfedder. Gaussian process machine learning and kriging for groundwater salinity interpolation. *Environmental Modelling & Software*, 144:105170, 2021.

[27] S. De Vito, G. Di Francia, E. Esposito, S. Ferlito, F. Formisano, and E. Massera. Adaptive machine learning strategies for network calibration of iot smart air quality monitoring devices. *Pattern Recognition Letters*, 136:264–271, 2020.

[28] F. Delaine, B. Lebental, and H. Rivano. In situ calibration algorithms for environmental sensor networks: A review. *IEEE Sensors Journal*, 19(15):5968–5978, 2019.

[29] R. J. Delfino, C. Sioutas, and S. Malik. Potential role of ultrafine particles in associations between airborne particle mass and cardiovascular health. *Environmental health perspectives*, 113(8):934–946, 2005.

[30] H. Ding and G. Noh. A hybrid model for spatiotemporal air quality prediction based on interpretable neural networks and a graph neural network. *Atmosphere*, 14(12):1807, 2023.

[31] G. I. Drewil and R. J. Al-Bahadili. Forecast air pollution in smart city using deep learning techniques: a review. *Multicultural Education*, 7 (5):38–47, 2021.

[32] D. Duvenaud, J. Lloyd, R. Grosse, J. Tenenbaum, and G. Zoubin. Structure discovery in nonparametric regression through compositional ker-

nel search. In *International Conference on Machine Learning*, pages 1166–1174. PMLR, 2013.

[33] V. P. Dwivedi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson. Graph neural networks with learnable structural and positional representations. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=wTTjnvGphYj`.

[34] M.-L. Eckert, K. Um, and N. Thuerey. Scalarflow: a large-scale volumetric data set of real-world scalar transport flows for computer animation and machine learning. *ACM Transactions on Graphics (TOG)*, 38 (6):1–16, 2019.

[35] D. Elfverson and C. Lejon. Use and scalability of openfoam for wind fields and pollution dispersion with building-and ground-resolving topography. *Atmosphere*, 12(9):1124, 2021.

[36] S. Emeis, K. Schafer, and C. Munkel. Observation of the structure of the urban boundary layer with different ceilometers and validation by rass data. *Meteorologische Zeitschrift*, 18(2):149, 2009.

[37] S. Emeis, K. Schäfer, C. Münkel, R. Friedl, and P. Suppan. Evaluation of the interpretation of ceilometer data with rass and radiosonde data. *Boundary-layer meteorology*, 143:25–35, 2012.

[38] H. Fan, S. Cheng, A. J. de Nazelle, and R. Arcucci. An efficient vit-based spatial interpolation learner for field reconstruction. In *International Conference on Computational Science*, pages 430–437. Springer, 2023.

[39] Y. Feng, J.-S. Kim, J.-W. Yu, K.-C. Ri, S.-J. Yun, I.-N. Han, Z. Qi, and X. Wang. Spatiotemporal informer: A new approach based on spatiotemporal embedding and attention for air quality forecasting. *Environmental Pollution*, 336:122402, 2023.

[40] C. Gariazzo, C. Silibello, S. Finardi, P. Radice, A. Piersanti, G. Calori, A. Cecinato, C. Perrino, F. Nussio, M. Cagnoli, et al. A gas/aerosol

air pollutants study over the urban area of rome using a comprehensive chemical transport model. *Atmospheric Environment*, 41(34):7286–7303, 2007.

[41] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[42] J. Han, H. Liu, H. Xiong, and J. Yang. Semi-supervised air quality forecasting via self-supervised hierarchical graph neural network. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):5230–5243, 2022.

[43] S. Han, W. Kundhikanjana, P. Towashiraporn, and D. Stratoulias. Interpolation-based fusion of sentinel-5p, srtm, and regulatory-grade ground stations data for producing spatially continuous maps of pm2. 5 concentrations nationwide over thailand. *Atmosphere*, 13(2):161, 2022.

[44] D. Hasenfratz, O. Saukh, and L. Thiele. On-the-fly calibration of low-cost gas sensors. In *European Conference on Wireless Sensor Networks*, pages 228–244. Springer, 2012.

[45] A. Heintzelman, G. M. Filippelli, M. J. Moreno-Madriñan, J. S. Wilson, L. Wang, G. K. Druschel, and V. O. Lulla. Efficacy of low-cost sensor networks at detecting fine-scale variations in particulate matter in urban environments. *International Journal of Environmental Research and Public Health*, 20(3):1934, 2023.

[46] B. Hoffmann, H. Boogaard, A. de Nazelle, Z. J. Andersen, M. Abramson, M. Brauer, B. Brunekreef, F. Forastiere, W. Huang, H. Kan, et al. Who air quality guidelines 2021–aiming for healthier air for all: a joint statement by medical, public health, scientific societies and patient representative organisations. *International journal of public health*, 66: 1604465, 2021.

[47] J. Hu, Y. Liang, Z. Fan, H. Chen, Y. Zheng, and R. Zimmermann.

Graph neural processes for spatio-temporal extrapolation. *arXiv preprint arXiv:2305.18719*, 2023.

[48] C. Hua, F. Berto, M. Poli, S. Massaroli, and J. Park. Learning efficient surrogate dynamic models with graph spline networks. *Advances in Neural Information Processing Systems*, 36:52523–52547, 2023.

[49] S. Janny, A. Beneteau, M. Nadri, J. Digne, N. Thome, and C. Wolf. Eagle: Large-scale learning of turbulent fluid dynamics with mesh transformers. In *International Conference on Learning Representation*, 2023.

[50] J. Janßen, P. Tremper, and T. Riedel. Adaptives luftqualitätsgewichtetes fahrradrouting mittels land-use regression auf basis offener daten. In *INFORMATIK 2021*, pages 321–331. Gesellschaft für Informatik, Bonn, 2021.

[51] A. P. Jeanjean, R. Buccolieri, J. Eddy, P. S. Monks, and R. J. Leigh. Air quality affected by trees in real street canyons: The case of marylebone neighbourhood in central london. *Urban Forestry & Urban Greening*, 22:41–53, 2017.

[52] S.-S. Jin. Compositional kernel learning using tree-based genetic programming for gaussian process regression. *Structural and Multidisciplinary Optimization*, 62(3):1313–1351, 2020.

[53] D. S. Jo, B. A. Nault, S. Tilmes, A. Gettelman, C. S. McCluskey, A. Hodzic, D. K. Henze, M. O. Nawaz, K. M. Fung, and J. L. Jimenez. Global health and climate effects of organic aerosols from different sources. *Environmental science & technology*, 57(37):13793–13807, 2023.

[54] S. Karagiannakos. Best graph neural networks architectures: Gcn, gat, mpnn and more. *https://theaisummer.com/*, 2021.

[55] M. Karl, S.-E. Walker, S. Solberg, and M. O. Ramacher. The eulerian urban dispersion model episode–part 2: Extensions to the source dispersion and photochemistry for episode–citychem v1. 2 and its application

to the city of hamburg. *Geoscientific Model Development*, 12(8):3357–3399, 2019.

[56] H. Kim and Y. W. Teh. Scaling up the automatic statistician: Scalable structure discovery using gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 575–584. PMLR, 2018.

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

[58] K. Klemmer, N. S. Safir, and D. B. Neill. Positional encoder graph neural networks for geographic data. In *International Conference on Artificial Intelligence and Statistics*, pages 1379–1389. PMLR, 2023.

[59] B. Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

[60] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi. Time series forecasting using a deep belief network with restricted boltzmann machines. *Neurocomputing*, 137:47–56, 2014.

[61] T. Lauriks, R. Longo, D. Baetens, M. Derudi, A. Parente, A. Bellemans, J. Van Beeck, and S. Denys. Application of improved cfd modeling for prediction and mitigation of traffic-related air pollution hotspots in a realistic urban street. *Atmospheric Environment*, 246:118127, 2021.

[62] C. Li, M. Budde, P. Tremper, T. Riedel, M. Beigl, et al. Smartaqnet 2020: A new open urban air quality dataset from heterogeneous pm sensors. *Proscience*, 8:1–10, 2021. ISSN 2283-5954.

[63] C. Li, M. Budde, P. Tremper, K. Schäfer, J. Riesterer, J. Redelstein, E. Petersen, M. Khedr, X. Liu, M. Köpke, et al. Smartaqnet 2020: a new open urban air quality dataset from heterogeneous pm sensors. *Proscience*, 8, 2022.

[64] J. Li, Y. Shen, L. Chen, and C. W. W. Ng. Rainfall spatial interpolation with graph neural networks. In *International Conference on Database Systems for Advanced Applications*, pages 175–191. Springer, 2023.

[65] J. Li, Y. Shen, L. Chen, and C. W. W. Ng. Ssin: Self-supervised learning for rainfall spatial interpolation. *Proceedings of the ACM on Management of Data*, 1(2):1–21, 2023.

[66] J. J. Li, B. Faltings, O. Saukh, D. Hasenfratz, and J. Beutel. Sensing the air we breathe—the opensense zurich dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 323–325, 2012.

[67] K. Li, Y. Liu, X. Ao, J. Chi, J. Feng, H. Yang, and Q. He. Reliable representations make a stronger defender: Unsupervised structure refinement for robust gnn. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 925–935, 2022.

[68] R. Li, S. Wang, F. Zhu, and J. Huang. Adaptive graph convolutional neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[69] Z. Li and A. B. Farimani. Graph neural network-accelerated lagrangian fluid simulation. *Computers & Graphics*, 103:201–211, 2022.

[70] S. Liang, T. Khalafbeigi, H. van Der Schaaf, B. Miles, K. Schleidt, S. Grellet, M. Beaufils, and M. Alzona. Ogc sensorthings api part 1: Sensing version 1.1. In *Open geospatial consortium*, 2021.

[71] H. Lin, S. Li, J. Xing, J. Yang, Q. Wang, L. Dong, and X. Zeng. Fusing retrievals of high resolution aerosol optical depth from landsat-8 and sentinel-2 observations over urban areas. *Remote Sensing*, 13(20):4140, 2021.

[72] J. Liu, Y. Chen, B. Ni, W. Ren, Z. Yu, and X. Huang. Fast fluid simulation via dynamic multi-scale gridding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1675–1682, 2023.

[73] M. Liu, A. Zeng, M. Chen, Z. Xu, Q. Lai, L. Ma, and Q. Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35: 5816–5828, 2022.

[74] N. Liu, X. Wang, L. Wu, Y. Chen, X. Guo, and C. Shi. Compact graph structure learning via mutual information compression. In *Proceedings of the ACM Web Conference 2022*, pages 1601–1610, 2022.

[75] Y. Liu, Y. Zheng, D. Zhang, H. Chen, H. Peng, and S. Pan. Towards unsupervised deep graph structure learning. In *Proceedings of the ACM Web Conference 2022*, pages 1392–1403, 2022.

[76] M. P. Lucas, R. J. Longman, T. W. Giambelluca, A. G. Frazier, J. Mclean, S. B. Cleveland, Y.-F. Huang, and J. Lee. Optimizing automated kriging to improve spatial interpolation of monthly rainfall over complex terrain. *Journal of Hydrometeorology*, 23(4):561–572, 2022.

[77] D. Luo, W. Cheng, W. Yu, B. Zong, J. Ni, H. Chen, and X. Zhang. Learning to drop: Robust graph neural network via topological denoising. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 779–787, 2021.

[78] J.-F. Markert, M. Budde, G. Schindler, M. Klug, and M. Beigl. Private rendezvous-based calibration of low-cost sensors for participatory environmental sensing. In *Proceedings of the Second International Conference on IoT in Urban Space*, pages 82–85, 2016.

[79] A. Martilli, B. Sánchez, J. L. Santiago, D. Rasilla, G. Pappaccogli, F. Allende, F. Martín, C. Roman-Cascón, C. Yagüe, and F. Fernández. Simulating the pollutant dispersion during persistent wintertime thermal inversions over urban areas. the case of madrid. *Atmospheric Research*, 270:106058, 2022.

[80] A. Masood and K. Ahmad. Data-driven predictive modeling of pm2. 5 concentrations using machine learning and deep learning techniques: a

case study of delhi, india. *Environmental Monitoring and Assessment*, 195(1):60, 2023.

[81] K. Meldrum, C. Guo, E. L. Marczylo, T. W. Gant, R. Smith, and M. O. Leonard. Mechanistic insight into the impact of nanomaterials on asthma and allergic airway disease. *Particle and Fibre Toxicology*, 14:1–35, 2017.

[82] Y. Miao, S. Liu, Y. Zheng, S. Wang, and Y. Li. Numerical study of traffic pollutant dispersion within different street canyon configurations. *Advances in meteorology*, 2014(1):458671, 2014.

[83] H. Montazeri and B. Blocken. Cfd simulation of wind-induced pressure coefficients on buildings with and without balconies: Validation and sensitivity analysis. *Building and environment*, 60:137–149, 2013.

[84] J. Muñoz-Sabater, E. Dutra, A. Agustí-Panareda, C. Albergel, G. Arduini, G. Balsamo, S. Boussetta, M. Choulga, S. Harrigan, H. Hersbach, et al. Era5-land: A state-of-the-art global reanalysis dataset for land applications. *Earth system science data*, 13(9):4349–4383, 2021.

[85] M. A. Njifon and D. Schuhmacher. Graph convolutional networks for spatial interpolation of correlated data. *Spatial Statistics*, 60:100822, 2024.

[86] NOAA. Aircraft based observation (abo) dataset, 2023. URL `https://madis.ncep.noaa.gov/madis_acars.shtml`.

[87] G. Oberdörster, Z. Sharp, V. Atudorei, A. Elder, R. Gelein, W. Kreyling, and C. Cox. Translocation of inhaled ultrafine particles to the brain. *Inhalation toxicology*, 16(6-7):437–445, 2004.

[88] B. T. Ong, K. Sugiura, and K. Zettsu. Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting pm2. 5. *Neural Computing and Applications*, 27(6):1553–1566, 2016.

[89] M. Pantusheva, R. Mitkov, P. O. Hristov, and D. Petrova-Antonova. Air pollution dispersion modelling in urban environment using cfd: a systematic review. *Atmosphere*, 13(10):1640, 2022.

[90] C. Peralta, H. Nugusse, S. Kokilavani, J. Schmidt, and B. Stoevesandt. Validation of the simplefoam (rans) solver for the atmospheric boundary layer in complex terrain. In *ITM Web of Conferences*, volume 2, page 01002. EDP Sciences, 2014.

[91] D. Petrova-Antonova, J. Jelyazkov, and I. Pavlova. Air quality monitoring platform with multiple data source support. *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, pages 1–17, 2021.

[92] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, and P. Battaglia. Learning mesh-based simulation with graph networks. In *International conference on learning representations*, 2020.

[93] R. Qadir, J. Schnelle-Kreis, G. Abbaszade, J. Arteaga-Salas, J. Diemer, and R. Zimmermann. Spatial and temporal variability of source contributions to ambient pm10 during winter in augsburg, germany using organic and inorganic tracers. *Chemosphere*, 103:263–273, 2014.

[94] D. Qin, J. Yu, G. Zou, R. Yong, Q. Zhao, and B. Zhang. A novel combined prediction scheme based on cnn and lstm for urban pm 2.5 concentration. *IEEE Access*, 7:20050–20059, 2019.

[95] N. I. Ramli, M. I. Ali, M. S. H. Saad, and T. Majid. Estimation of the roughness length (zo) in malaysia using satellite image. *Wind Engineering*, 2009.

[96] E. Rivas, J. L. Santiago, Y. Lechón, F. Martín, A. Ariño, J. J. Pons, and J. M. Santamaría. Cfd modelling of air quality in pamplona city (spain): Assessment, stations spatial representativeness and health impacts valuation. *Science of the Total Environment*, 649:1362–1380, 2019.

[97] D. R. Roberts, V. Bahn, S. Ciuti, M. S. Boyce, J. Elith, G. Guillera-Arroita, S. Hauenstein, J. J. Lahoz-Monfort, B. Schröder, W. Thuiller,

et al. Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40(8):913–929, 2017.

[98] T. K. Rusch, M. M. Bronstein, and S. Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023.

[99] K. A. Rychlik, J. R. Secrest, C. Lau, J. Pulczinski, M. L. Zamora, J. Leal, R. Langley, L. G. Myatt, M. Raju, R. C.-A. Chang, et al. In utero ultrafine particulate matter exposure causes offspring pulmonary immunosuppression. *Proceedings of the National Academy of Sciences*, 116(9): 3443–3448, 2019.

[100] B. Sanchez, J. L. Santiago, A. Martilli, F. Martin, R. Borge, C. Quaassdorff, and D. de la Paz. Modelling nox concentrations through cfd-rans in an urban hot-spot using high resolution traffic emissions and meteorology from a mesoscale model. *Atmospheric Environment*, 163:155–165, 2017.

[101] K. Schäfer, M. Elsasser, J. Arteaga-Salas, J. Gu, M. Pitz, J. Schnelle-Kreis, J. Cyrys, S. Emeis, A. Prevot, and R. Zimmermann. Source apportionment and the role of meteorological conditions in the assessment of air pollution exposure due to urban emissions. *Atmospheric Chemistry and Physics Discussions*, 14(2):2235–2275, 2014.

[102] K. Schäfer, K. Lande, H. Grimm, G. Jenniskens, R. Gijsbers, V. Ziegler, M. Hank, and M. Budde. High-resolution assessment of air quality in urban areas—a business model perspective. *Atmosphere*, 12(5):595, 2021.

[103] R. Schlund, J. Riesterer, M. Köpke, M. Kowalski, P. Tremper, M. Budde, and M. Beigl. Calibration of low-cost particulate matter sensors with elastic weight consolidation (ewc) as an incremental deep learning method. In *International Summit Smart City 360°*, pages 596–614. Springer, 2020.

[104] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. part b. on the problem of oscu-

latory interpolation. a second class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4(2):112–141, 1946.

[105] C. C. C. Service and C. D. Store. Global marine surface meteorological variables from 1851 to 2010 from comprehensive in-situ observations. Copernicus Climate Change Service (C3S) Climate Data Store (CDS), 2021. URL `https://cds.climate.copernicus.eu/cdsapp#!/dataset/10.24381/cds.27f643d7`. DOI: 10.24381/cds.27f643d7.

[106] Y. Shen, S. Lehmler, S. M. Murshed, and T. Riedel. Characterizing air quality in urban areas with mobile measurement and high resolution open spatial data: Comparison of different machine-learning approaches using a visual interface. In *Science and Technologies for Smart Cities: 5th EAI International Summit, SmartCity360, Braga, Portugal, December 4-6, 2019, Proceedings*, pages 115–126. Springer, 2020.

[107] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524, 1968.

[108] K. P. Singh, S. Gupta, A. Kumar, and S. P. Shukla. Linear and nonlinear modeling approaches for urban air quality prediction. *Science of the Total Environment*, 426:244–255, 2012.

[109] F. Slemr, G. Baumbach, P. Blank, U. Corsmeier, F. Fiedler, R. Friedrich, M. Habram, N. Kalthoff, D. Klemp, J. Kühlwein, et al. Evaluation of modeled spatially and temporarily highly resolved emission inventories of photosmog precursors for the city of augsburg: the experiment eva and its major results. *Tropospheric Chemistry: Results of the German Tropospheric Chemistry Programme*, pages 207–233, 2002.

[110] L. H. Slørdal, S.-E. Walker, and S. Solberg. The urban air dispersion model episode applied in airquis2003. technical description. *NILU TR*, 2003.

[111] E. G. Snyder, T. H. Watkins, P. A. Solomon, E. D. Thoma, R. W. Williams, G. S. Hagler, D. Shelow, D. A. Hindin, V. J. Kilaru, and P. W.

Preuss. The changing paradigm of air pollution monitoring. *Environmental science & technology*, 47(20):11369–11377, 2013.

[112] I.-F. Su, Y.-C. Chung, C. Lee, and P.-M. Huang. Effective pm2. 5 concentration forecasting based on multiple spatial–temporal gnn for areas without monitoring stations. *Expert Systems with Applications*, 234: 121074, 2023.

[113] J. Su, L. Wang, Z. Gu, M. Song, and Z. Cao. Effects of real trees and their structure on pollutant dispersion and flow field in an idealized street canyon. *Atmospheric Pollution Research*, 10(6):1699–1710, 2019.

[114] S. Suganya and T. Meyyappan. Adaptive deep learning model for air pollution analysis using meteorological big data. In *2021 2nd International Conference on Communication, Computing and Industry 4.0 (C2I4)*, pages 1–6. IEEE, 2021.

[115] S. Sun, G. Zhang, C. Wang, W. Zeng, J. Li, and R. Grosse. Differentiable compositional kernel learning for gaussian processes. In *International Conference on Machine Learning*, pages 4828–4837. PMLR, 2018.

[116] C. Tee, E. Ng, and G. Xu. Analysis of transport methodologies for pollutant dispersion modelling in urban environments. *Journal of Environmental Chemical Engineering*, 8(4):103937, 2020.

[117] W. Tobler. On the first law of geography: A reply. *Annals of the Association of American Geographers*, 94:304–310, 2004.

[118] W. R. Tobler. A computer movie simulating urban growth in the detroit region. *Economic geography*, 46(sup1):234–240, 1970.

[119] Y. Tominaga and T. Stathopoulos. Turbulent schmidt numbers for cfd analysis with various types of flowfield. *Atmospheric Environment*, 41 (37):8091–8099, 2007.

[120] P. Tremper, T. Riedel, and M. Budde. Spatial interpolation of air quality data with multidimensional gaussian processes. In *INFORMATIK 2021*, pages 269–286. Gesellschaft für Informatik, Bonn, 2021.

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

[122] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[123] G. Wan and H. Kokel. Graph sparsification via meta-learning. *DLG@ AAAI*, 2021.

[124] H. Wang, Y. Cao, Z. Huang, Y. Liu, P. Hu, X. Luo, Z. Song, W. Zhao, J. Liu, J. Sun, et al. Recent advances on machine learning for computational fluid dynamics: A survey. *arXiv preprint arXiv:2408.12171*, 2024.

[125] H. Wang, J. Li, A. Dwivedi, K. Hara, and T. Wu. Beno: Boundary-embedded neural operators for elliptic pdes. *arXiv preprint arXiv:2401.09323*, 2024.

[126] R. Wang, K. Kashinath, M. Mustafa, A. Albert, and R. Yu. Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1457–1466, 2020.

[127] R. Wang, R. Walters, and R. Yu. Incorporating symmetry into deep dynamics models for improved generalization. *arXiv preprint arXiv:2002.03061*, 2020.

[128] X. Wei, N.-B. Chang, K. Bai, and W. Gao. Satellite remote sensing of aerosol optical depth: advances, challenges, and perspectives. *Critical Reviews in Environmental Science and Technology*, 50(16):1640–1725, 2020.

[129] H. Wessels, C. Weißenfels, and P. Wriggers. The neural particle method–an updated lagrangian physics informed neural network for computational fluid dynamics. *Computer Methods in Applied Mechanics and Engineering*, 368:113127, 2020.

[130] P. with Code. Gcn explained, 2023. URL https://paperswithcode.com/method/gcn. Accessed: 2025-01-29.

[131] Q. Wu, W. Zhao, Z. Li, D. P. Wipf, and J. Yan. Nodeformer: A scalable graph structure learning transformer for node classification. *Advances in Neural Information Processing Systems*, 35:27387–27401, 2022.

[132] Y. Wu, D. Zhuang, A. Labbe, and L. Sun. Inductive graph neural networks for spatiotemporal kriging. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4478–4485, 2021.

[133] Y. Wu, D. Zhuang, M. Lei, A. Labbe, and L. Sun. Spatial aggregation and temporal convolution networks for real-time kriging. *arXiv preprint arXiv:2109.12144*, 2021.

[134] H. Xu, L. Xiang, J. Yu, A. Cao, and X. Wang. Speedup robust graph structure learning with low-rank information. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2241–2250, 2021.

[135] X. Xu, S. Hu, H. Shao, P. Shi, R. Li, and D. Li. A spatio-temporal forecasting model using optimally weighted graph convolutional network and gated recurrent unit for wind speed of different sites distributed in an offshore wind farm. *Energy*, 284:128565, 2023.

[136] Y. Yang and M. Jia. 3d spatial interpolation of soil heavy metals by combining kriging with depth function trend model. *Journal of Hazardous Materials*, 461:132571, 2024.

[137] Y. Yang, J. Cermak, X. Chen, Y. Chen, and X. Hou. High-resolution pm10 estimation using satellite data and model-agnostic meta-learning. *Remote Sensing*, 16(13):2498, 2024.

[138] A. W. M. Yazid, N. A. C. Sidik, S. M. Salim, and K. M. Saqr. A review on the flow structure and pollutant dispersion in urban street canyons for urban planning strategies. *Simulation*, 90(8):892–916, 2014.

[139] Y. Ye, S. Zhang, and J. J. Yu. Spatial-temporal traffic data imputation via graph attention convolutional network. In *International Conference on Artificial Neural Networks*, pages 241–252. Springer, 2021.

[140] D. Yu, R. Zhang, Z. Jiang, Y. Wu, and Y. Yang. Graph-revised convolutional network. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part III*, pages 378–393. Springer, 2021.

[141] M. Yu, A. Masrur, and C. Blaszczak-Boxe. Predicting hourly pm2. 5 concentrations in wildfire-prone areas using a spatiotemporal transformer model. *Science of The Total Environment*, 860:160446, 2023.

[142] N. Zaini, L. W. Ean, A. N. Ahmed, M. Abdul Malek, and M. F. Chow. Pm2. 5 forecasting for an urban area based on deep learning and decomposition method. *Scientific Reports*, 12(1):17565, 2022.

[143] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah. Data augmentation for graph neural networks. In *Proceedings of the aaai conference on artificial intelligence*, volume 35, pages 11015–11023, 2021.

[144] Z. Zhiyao, S. Zhou, B. Mao, X. Zhou, J. Chen, Q. Tan, D. Zha, Y. Feng, C. Chen, and C. Wang. Opengsl: A comprehensive benchmark for graph structure learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[145] 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*, volume 35, pages 11106–11115, 2021.

[146] Y. Zhu, Y. Xu, F. Yu, S. Wu, and L. Wang. Cagnn: Cluster-aware graph neural networks for unsupervised graph representation learning. *arXiv preprint arXiv:2009.01674*, 2020.

[147] Y. Zhu, W. Xu, J. Zhang, Y. Du, J. Zhang, Q. Liu, C. Yang, and S. Wu. A survey on graph structure learning: Progress and opportunities. *arXiv preprint arXiv:2103.03036*, 2021.

[148] Y. Zhu, W. Xu, J. Zhang, Q. Liu, S. Wu, and L. Wang. Deep graph structure learning for robust representations: A survey. *arXiv preprint arXiv:2103.03036*, 14:1–1, 2021.

[149] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu. *The finite element method: its basis and fundamentals*. Elsevier, 2005.

[150] D. Zou, H. Peng, X. Huang, R. Yang, J. Li, J. Wu, C. Liu, and P. S. Yu. Se-gsl: A general and effective graph structure learning framework through structural entropy optimization. In *Proceedings of the ACM Web Conference 2023*, pages 499–510, 2023.