POLITECNICO MILANO 1863

Graph Neural Networks for IoT Sensor Network Optimization: A Spatio-Temporal Forecasting Approach

International Master's in Business Analytics and Data Science

Author: Swathi Kalburgi
Advisor: Prof. Mattia Cerutti
Academic Year: 2024 - 2025

Abstract

The proliferation of Internet of Things (IoT) deployments has created vast sensor networks that generate complex spatiotemporal data streams. Traditional forecasting methods treat sensors as independent entities, failing to capture the spatial dependencies inherent in networked systems. This thesis investigates the application of **Graph Neural Networks (GNNs)**, specifically **Graph Convolution Networks (GCNs)**, for temperature forecasting in IoT sensor networks.

Using the Intel Berkeley Research Lab dataset containing 2.3 million readings from 54 sensors, a 5-nearest neighbor graph was constructed, and a two-layer GCN was implemented with sliding-window temporal sequences. After rigorous preprocessing, including removal of sensors with excessive missing data, outlier filtering, and per-sensor normalization, the model was trained to forecast temperature values across multiple future time steps using historical observations.

The optimized GCN demonstrated substantial performance improvements over classical baselines, achieving **73.16% better accuracy** than the best traditional method. The model showed strong predictive capability with high correlation between predictions and actual values, while maintaining minimal systematic bias. Ablation studies confirmed temperature as the most influential input feature, while spatial analysis revealed performance heterogeneity across sensors based on their network position. Centrally-located sensors benefited from richer neighborhood context, while peripherally-located sensors exhibited higher errors due to limited connectivity and exposure to boundary effects.

Horizon-wise evaluation showed stable performance across multi-step forecasts, indicating robust short-term prediction capability. These findings demonstrate that GNNs effectively leverage network topology for superior forecasting performance in distributed sensor systems, with significant implications for predictive maintenance, energy optimization, and intelligent building management.

Keywords: Graph Neural Networks, IoT Sensor Networks, Spatio-Temporal Forecasting, Temperature Prediction, Graph Convolution Networks, Intel Berkeley Lab Dataset, Anomaly Detection

Executive Summary: Graph Neural Networks achieve better temperature forecasting accuracy than classical methods by explicitly modeling spatial dependencies between IoT sensors, enabling predictive building climate control and early fault detection.

Table of Contents

Chapter 1: Introduction

1.1. Context and Motivation

The Internet of Things (IoT) underpins a growing share of modern infrastructure, enabling real-time monitoring and closed-loop control across buildings, factories, cities, and environmental systems. These deployments generate continuous multivariate streams, most commonly temperature, humidity, light, and voltage, captured by spatially distributed sensors. By 2034, global IoT connections are projected to exceed **40 billion** devices, underscoring both the scale and operational importance of accurate, robust forecasting in networked settings (Statista, 2025)[1].

A defining property of such data is **spatial interdependence**: sensors located near one another tend to co-evolve because they share airflows, radiant heat, occupancy patterns, and sometimes power lines or control zones. In practice, temperature in one office often anticipates or mirrors temperature in the adjacent corridor; conversely, sensors separated by partitions or exterior boundaries experience different dynamics and time lags. Traditional forecasting approaches, such as ARIMA and Random Forest, as well as sequence models like LSTMs, typically treat each sensor as an isolated time series. This independence assumption disregards neighborhood signals that are relevant in real buildings.

The **Intel Berkeley Research Lab Sensor Dataset** [2] used in this thesis exemplifies these realities. It contains **~2.3 million readings** from **54 sensors**, each recording temperature, humidity, light, and voltage at regular intervals. Exploratory analysis confirms strong, uneven correlations across the floor plan: sensors sharing corridors or air streams move together, while those separated by walls exhibit weaker or phase-shifted responses.

Why spatial dependencies matter in practice.
Spatial coupling determines how local disturbances propagate. Warm air from a crowded meeting room gradually influences nearby spaces; exterior walls and open boundaries introduce variability that central areas do not experience. Traditional models that treat sensors independently cannot capture the coupling, leading to three critical failures:

1. **Information loss**: Ignoring spatial structure discards valuable predictive signals from neighboring sensors.
2. **Inefficient learning**: Models must independently learn patterns that are actually shared across spatially proximate sensors.
3. **Poor generalization**: Without understanding network topology, models struggle when sensor configurations change or sensors fail.

Fig. 1 displays the **Intel Lab Sensor Network Layout**, where each orange marker represents a sensor node plotted by its physical coordinates (x, y in meters). The irregular distribution, characterized by dense clusters in central areas and sparse placement near the peripheries, accurately reflects real deployment constraints and directly influences the connectivity pattern used for graph construction. Regions of sparse coverage correspond to weaker neighborhood influence, making those nodes more error-prone during forecasting.

Fig. 2 illustrates the **Sensor × Horizon Error Heatmap ($\log_{10}$ MSE)** for the 20 worst-performing sensors across three forecast horizons. This visualization reveals the spatial heterogeneity of forecasting difficulty within the network: sensors located near external boundaries (e.g., Sensor 45) tend to produce higher error values, while centrally positioned sensors (e.g., Sensor 18) achieve lower mean errors due to richer neighborhood information.

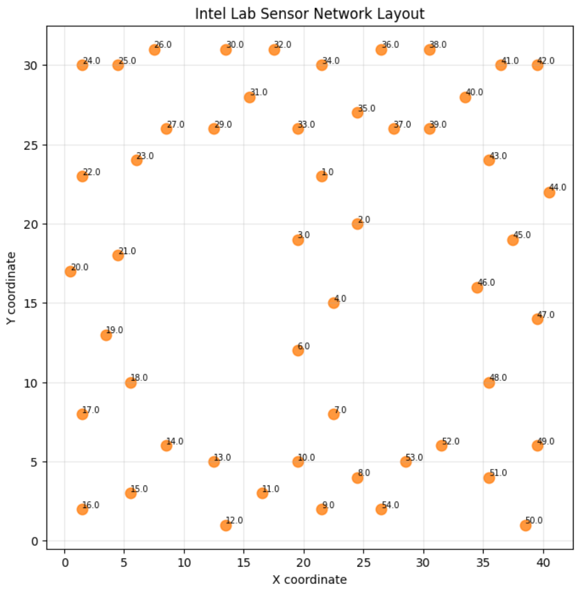


Fig. 1: Intel Lab Sensor Network Layout

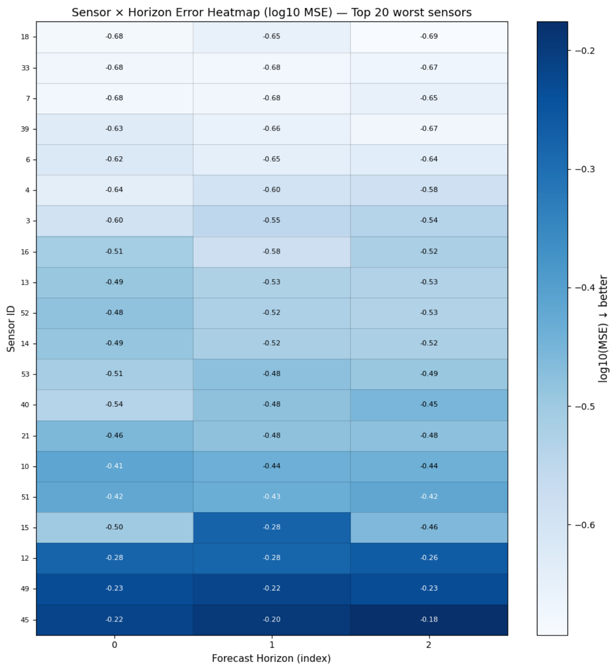


Fig. 2: Sensor × Horizon Error Heatmap ($\log_{10}$ MSE)

Understanding Fig. 2: Each cell shows $\log_{10}$(MSE), where more negative values indicate better performance (lower error). The vertical axis lists sensors, ordered from worst (top) to best (bottom), among the top 20 worst performers. The horizontal axis shows forecast horizon (0, 1, 2 corresponding to 1-step, 2-step, 3-step ahead). The color gradient from light (worse) to dark (better) makes performance patterns immediately visible. The heatmap reveals three key patterns:

1. **Boundary effects**: Sensors at network edges consistently show lighter colors (higher errors) because they have fewer neighbors to draw information from and are more exposed to external environmental fluctuations.
2. **Horizon consistency**: Most sensors show relatively stable performance across horizons, indicating that prediction difficulty is primarily sensor-dependent rather than horizon-dependent.
3. **Error clustering**: Adjacent sensors in the network topology often appear in similar positions in this ranking, suggesting that local environmental conditions (ventilation patterns, sunlight exposure) affect entire neighborhoods.

These spatial variations are not random fluctuations but manifestations of underlying **physical causality**. Edge sensors experience higher variance because of exposure to open boundaries or external airflow, whereas interior sensors benefit from stable, thermally buffered surroundings. This structural heterogeneity directly motivated the adoption of **GNNs**, which explicitly model such inter-sensor relationships by representing sensors as nodes and their spatial or statistical proximities as edges.
From a practical standpoint, enhancing forecasting accuracy in IoT systems yields tangible real-world benefits:

- **Energy optimization:** Smart HVAC control strategies have been reported to reduce energy use by 10–25 % in building studies (El Husseini et al., 2025) [3].
- **Predictive maintenance:** Early fault detection from spatially correlated anomalies prevents costly equipment failures.

- **Occupant comfort:** Adaptive climate regulation responds more effectively to localized environmental shifts.
- **Fault diagnosis:** Graph-based anomaly clustering helps isolate malfunctioning sensors or environmental shifts.

By integrating both spatial and temporal dependencies, **GNNs** provide a foundation for models that are not only more accurate but also **interpretable**, enabling data-driven decisions grounded in physical network structure rather than isolated statistical trends.

1.2. Problem Statement

Despite progress in deep learning and time-series analysis, existing forecasting approaches fall short when applied to **non-Euclidean sensor networks**. The limitations can be summarized in three main points:

1. **Spatial blindness:** Classical statistical models (e.g., Linear Regression, ARIMA) and temporal deep networks (RNNs, LSTMs) treat each sensor's sequence in isolation. They fail to account for **neighborhood influence**, such as how heat diffusion or shared airflow links adjacent nodes. Without spatial context, localized fluctuations appear as random noise rather than system-wide patterns.
2. **Topology ignorance:** IoT sensors are deployed irregularly, clustered in dense zones and sparse near peripheries. Grid-based methods like CNNs assume uniform spacing and thus distort adjacency when applied to irregular layouts. **GNNs** overcome this by representing sensors as nodes and spatial or statistical proximities as edges, preserving the real network geometry (Kipf & Welling, 2017) [4].
3. **Robustness fragility:** Real-world IoT deployments face frequent **data missingness, link failures,** and **node dropout**. Traditional models degrade rapidly under these conditions because they cannot leverage redundancy from neighboring sensors to compensate for missing data.

Fig. 3 illustrates the missing-value distribution across all sensors in the Intel dataset. The heatmap reveals severe heterogeneity: while most sensors have <10% missing data, six sensors exhibit >50% missingness. This irregular sampling pattern introduces problems such as:

- **Temporal discontinuity**: Time-series models like ARIMA require evenly spaced observations. Gaps create artificial patterns in autocorrelation.
- **Biased statistics**: Computing mean/variance from incomplete data underestimates true variability if the missingness is non-random.
- **Training instability**: Neural networks struggle when some sensors contribute 50,000 samples while others contribute 5000.

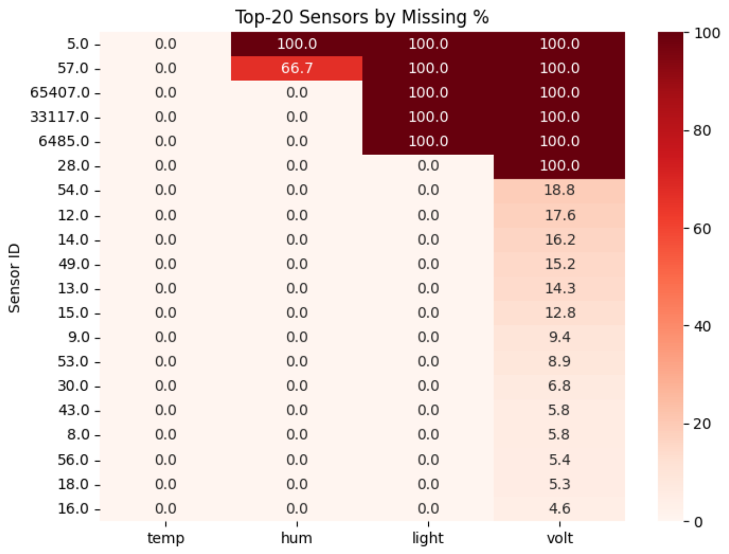


Fig. 3: Missing data heatmap showing irregular sensor completeness

These data quality challenges compound the fundamental limitation of spatial ignorance. Traditional forecasting methods provide no principled way to handle sensors with extensive missing data because they treat each sensor independently. In contrast, graph-based approaches can propagate information from well-connected, reliable sensors to compensate for poorly-performing or incomplete nodes through the network structure.

The Intel Lab dataset confirms strong inter-feature dependencies: temperature-humidity correlation (r = -0.505) and temperature-light correlation (r = 0.358) demonstrate that features don't evolve independently. This multivariate coupling further motivates graph-aware modeling that can capture both spatial and feature-level relationships.

The Central Challenge
The core problem this thesis addresses is whether explicitly modeling network topology through GNNs can overcome these limitations and achieve superior forecasting performance compared to traditional methods that ignore spatial structure. Specifically, can a graph-based approach that represents sensors as nodes and their spatial relationships as edges capture the neighborhood influence, handle irregular deployment patterns, and maintain robustness to data quality issues that plague conventional forecasting methods?

This question has direct practical implications. If GNNs can leverage spatial dependencies to improve prediction accuracy significantly, they would enable more effective building management systems, reduce energy consumption through better predictive control, and support maintenance strategies that identify sensor or equipment failures before they cascade into costly system-wide problems.

1.3. Research Questions

RQ1: Performance Superiority
Can Graph Convolution Networks (GCNs) outperform traditional methods by explicitly modeling spatial dependencies in IoT sensor networks?

RQ2: Feature and Configuration Influence
Which input features (temperature, humidity, light, voltage) and graph configurations most influence forecasting performance?

RQ3: Spatial Performance Heterogeneity
How does prediction accuracy vary across sensors with differing spatial positions, data completeness, and neighbor connectivity?

RQ4: Practical Deployment Principles
What practical design principles can guide the deployment of GCN-based forecasting models in real-world IoT systems?

These questions collectively evaluate both the **predictive superiority** and **practical applicability** of topology-aware learning.

1.4. Research Contributions

This study makes the following contributions to the intersection of GNNs and IoT sensor forecasting:

1. **Methodological Framework:** A complete pipeline for transforming multivariate IoT sensor readings into graph-structured inputs, including:
    - K-nearest-neighbor graph construction with inverse-distance edge weights.
    - Sliding-window temporal segmentation.
    - Per-sensor z-score normalization that preserves within-sensor dynamics while enabling cross-sensor comparison.
2. **Empirical Validation:** Experimental evaluation demonstrating substantial performance improvements over traditional baselines. The GCN architecture achieved significantly better accuracy compared to classical machine learning methods, with strong linear agreement between predictions and ground truth. Statistical significance was confirmed through rigorous testing, and ablation studies quantified the relative importance of each input feature, identifying temperature as the dominant predictive variable.
3. **Spatial Diagnostics:** Detailed per-sensor and per-horizon analysis revealing systematic performance patterns across the network. The study showed considerable variation in prediction accuracy based on sensor location, with central sensors consistently outperforming peripheral ones. Multi-step forecast evaluation demonstrated stable short-term prediction capability, with minimal systematic bias across all sensors and horizons, confirming unbiased estimation throughout the network.
4. **Practical Insights:** Operational recommendations, including:
    - Optimal train/validation/test split (70/15/15) balances learning capacity and generalization.
    - Feature selection guidance for resource-constrained environments.
    - Special handling requirements for edge sensors due to limited neighborhood information.
    - Hyperparameter configurations that achieve optimal performance while maintaining computational efficiency.

Together, these contributions advance both the theoretical and applied understanding of GNN-based forecasting for sensor networks, providing a foundation for practical deployment in building management systems.

1.5. Thesis Structure

The remainder of this thesis is organized as follows:

- **Chapter 2** reviews related work on IoT forecasting, covering traditional, deep-learning, and graph-based models.
- **Chapter 3** explains the methodology, including the dataset preprocessing, graph construction, GCN architecture, and evaluation metrics.
- **Chapter 4** presents experimental results and baseline comparisons.
- **Chapter 5** analyzes forecasting performance across sensors and horizons, identifying best and worst performers.
- **Chapter 6** discusses findings, interpretability, and limitations.
- **Chapter 7** concludes with synthesized insights, answers to the research questions, and directions for future work.

Chapter 2: Literature Review

2.1. IoT Sensor Networks: Context and Challenges

The IoT has transformed how we monitor and manage physical environments, creating vast networks of interconnected devices that continuously generate data. In building automation systems specifically, IoT deployments have evolved from simple temperature monitoring to sophisticated multi-sensor networks that inform HVAC control, occupancy detection, and energy optimization (Gubbi et al., 2013)[5]. These systems present unique forecasting challenges that distinguish them from traditional time-series problems.

Architectural Characteristics
Real-world IoT sensor networks exhibit three defining properties that complicate forecasting:

- **Spatial heterogeneity**: Sensors are deployed according to practical constraints (power availability, coverage requirements, installation costs) rather than uniform grids. This creates irregular network topologies where connectivity varies across regions.
- **Temporal irregularity**: While ideal systems sample at constant intervals, real deployments experience communication delays, buffer overflows, and clock drift that create uneven sampling (Elsts et al., 2018) [6].
- **Multivariate coupling**: Sensors record multiple environmental variables (temperature, humidity, light, voltage) that exhibit complex interdependencies rather than evolving independently.

Data Quality Challenges
Three critical data quality issues emerge consistently in IoT deployments:

- **Missing data**: Hardware failures, communication errors, and power outages create gaps in sensor readings. Missingness patterns are often heterogeneous; some sensors may be highly reliable while others fail frequently.
- **Outliers**: Sensor malfunctions produce physically impossible readings: temperatures exceeding boiling point in an office building, negative humidity values, and voltage spikes during communication errors.
- **Scale heterogeneity**: Different sensor types use different scales and units. Temperature sensors may report in Celsius or Fahrenheit, light sensors in lux or arbitrary units, requiring careful normalization to enable cross-sensor learning.

Energy and Performance Trade-offs
Building energy consumption forecasting has garnered particular attention due to its economic and environmental impact. Buildings account for approximately 40% of global energy consumption, with HVAC systems representing the largest controllable load (Pérez-Lombard et al., 2008)[7]. Improving forecast accuracy by even 5-10% can yield substantial energy savings through more efficient predictive control strategies. However, IoT devices operate under strict energy constraints: wireless sensor nodes typically run on batteries, creating tension between sampling frequency, computational complexity, and operational lifetime.

These challenges motivate the need for forecasting methods that are simultaneously **robust to irregular data, capable of leveraging spatial structure**, and **computationally efficient** for deployment on resource-constrained devices.

2.2. Traditional Statistical Methods

Statistical time-series forecasting has dominated sensor data analysis for decades. ARIMA models decompose series into autoregressive components, differencing operations, and moving average terms, with parameters selected through diagnostic tests and information criteria (Box et al., 2016)[8]. The appeal is interpretability: each coefficient has a meaning, and diagnostic plots reveal model adequacy. Modern implementations provide accessible tools for seasonal decomposition and exponential smoothing (Hyndman & Athanasopoulos, 2021)[9].

Limitations for Multi-Sensor Networks

Vector Autoregression attempts to model multiple time series jointly, estimating how each sensor's current value depends on lagged values of all sensors in the network. For an N-sensor network, a first-order VAR model requires estimating N^2 parameters, one coefficient for how each sensor influences every other sensor. This creates two problems:

- **Computational expense**: Parameter estimation scales due to matrix inversion operations.
- **Statistical instability**: With limited training data, the model becomes overparameterized and overfits.

Regularization offers partial solutions. LASSO regression applies L1 penalties that drive many coefficients to exactly zero, producing sparse models (Tibshirani, 1996)[10]. However, LASSO cannot encode spatial structure; it doesn't "know" which sensors are neighbors. Practitioners must manually create features encoding neighborhood relationships, a brittle process that fails when network topology changes.

2.3. Machine Learning Baselines

Ensemble methods brought nonlinear modeling to sensor forecasting without requiring explicit functional form specification. Random Forests recursively partition the feature space using decision trees, with final predictions averaged across hundreds of trees to reduce variance (Breiman, 2001)[11]. The scikit-learn implementation enables efficient parallel training across CPU cores (Pedregosa et al., 2011)[12].

Random Forest's key limitation is architectural: it treats spatial relationships as independent input features rather than structured topology. To predict sensor i's temperature, the model receives lagged temperatures from neighboring sensors as separate columns in the feature matrix. The algorithm cannot learn that proximity matters.

Support Vector Machines with radial basis function kernels can capture complex nonlinear patterns but scale poorly, making them impractical for datasets exceeding a certain number of observations.

2.4. Deep Learning for Sequential Data

Long Short-Term Memory networks revolutionize time-series forecasting by learning temporal dependencies automatically. LSTMs maintain cell states that propagate information across arbitrary time horizons, avoiding the vanishing gradient problem that plagued earlier recurrent architectures. Applied to building energy prediction, LSTMs achieved 15-30% error reduction versus classical methods, with greatest gains during periods of rapid load changes (Fan et al., 2017)[13].

Architectural Mismatch with Sensor Networks

Despite these successes, LSTMs face a fundamental problem: they process sequences one timestamp at a time, assuming temporal ordering is meaningful. For sensor networks, spatial relationships are simultaneous: sensor i's neighbors influence it concurrently, not sequentially. Standard LSTMs architectures require artificially ordering sensors (process sensor 1, then 2, then 3…), but this ordering is arbitrary and discards the true graph structure.

2.5. Graph Neural Networks

GNNs emerged from recognizing that convolution, the operation underlying CNNs' success on images, can be generalized to irregular graphs. The foundational insight comes from spectral graph theory: just as images have frequency representations, graphs have spectral decompositions.

Core Message-Passing Framework
The seminal GCN paper introduced a simplified spectral convolution that operates efficiently on large graphs (Kipf & Welling, 2017)[4]. The core operation is intuitive: each node aggregates feature vectors from its neighbors, applies a learned linear transformation, and passes the result through a nonlinearity. This operation is:

- **Permutation-invariant**: The order we process neighbors doesn't affect the result.
- **Parameter-efficient**: The same weight matrix applies to all nodes.
- **Scalable**: Computation is linear in the number of edges.

A comprehensive taxonomy categorizes GNN architectures into four families: recurrent (iterate message passing to convergence), convolution (fixed-depth aggregation), attention-based (learned neighbor weighting), and spatial-temporal (combine graph structure with time-series modeling) (Wu et al., 2020)[14]. The survey notes that while GNNs excel at static graph tasks (node classification, link prediction), temporal forecasting requires explicit integration of sequential modeling.

2.6. Spatial-Temporal Graph Neural Networks

The integration of graph convolutions with temporal modeling unlocked forecasting on network-structured data. The three architectural paradigms that emerged are:

Spatial-Temporal Graph Convolution Networks (STGCN)
Yu et al. (2018)[15] introduced ST-blocks that sandwich graph convolution between temporal convolutions, creating layers that capture spatial dependencies within each timestep and temporal evolution across timesteps. Fig. 4 illustrates this spatial blocks performing graph convolutions to aggregate neighbor information, while temporal blocks apply 1-dimensional convolutions across time.

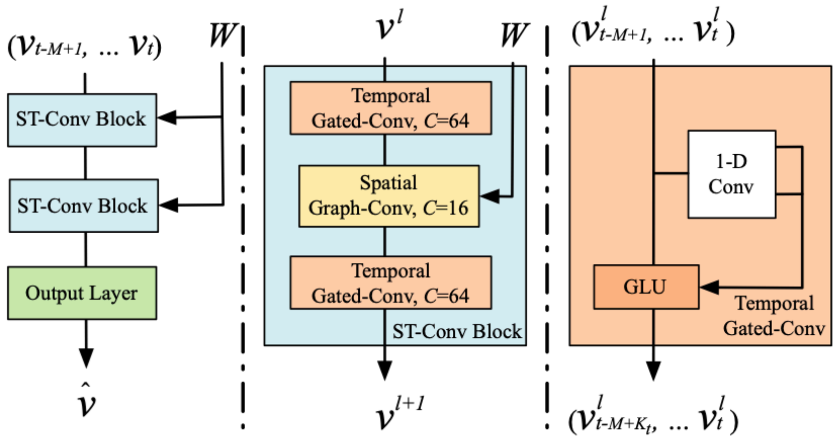


Fig. 4: STGCN architecture showing alternating spatial and temporal layers

Diffusion Convolution Recurrent Neural Networks (DCRNN)
Li et al. (2018)[16] replace standard RNN operations with graph diffusion, explicitly modeling how information propagates through networks. The intuition is physical; traffic congestion diffuses through

road networks, heat spreads through buildings, and signals propagate across sensor arrays. By parameterizing diffusion as a graph operation, DCRNN achieved state-of-the-art traffic forecasting while maintaining interpretability; learned diffusion coefficients reveal how disturbances propagate spatially.

Graph WaveNet
Wu et al. (2019)[17] introduced adaptive adjacency matrices learned jointly with predictions. While transportation networks have physical connectivity, latent traffic light synchronization, and shared congestion bottlenecks, they also influence flow. By allowing the model to discover these hidden connections, Graph EaveNet improved robustness and generalization.

2.7. Comparative Analysis

Table 1 synthesizes key trade-offs across forecasting paradigms examined in this review.

Approach	Spatial Modeling	Topology Awareness	Data Requirements	Interpretability
ARIMA/VAR	None (per-sensor)	No	Low	High
Random Forest	Feature engineering	No	Moderate	Moderate
LSTM/RNN	Sequential only	No	High	Low
CNN	Grid-based	Partial	High	Low
GCN	**Graph convolution**	**Yes**	**Moderate**	**Moderate**
STGCN/DCRNN	Graph + recurrent	Yes	Very high	Low

Table 1: Comparison of forecasting approaches

The critical distinction is **topology awareness**. Only graph-based methods explicitly represent network structure, enabling them to leverage spatial dependencies that other approaches ignore or awkwardly encode through manual feature engineering.

2.8. Research Gap and Thesis Positioning

This literature review reveals a clear progression: traditional methods ignore spatial structure → deep learning automates feature extraction but assumes grids/sequences → GNNs handle graphs but require temporal integration → STGNNs combine both, facilitating traffic forecasting.

IoT sensors differ from traffic systems in ways that challenge existing STGNN architectures:

- **Data quality**: Building sensors experience higher missing data rates and sensor failures than traffic systems.
- **Scale**: 50-100 nodes versus 1000+ nodes in traffic networks.
- **Multivariate features**: Temperature, humidity, light, voltage versus univariate speed/flow.
- **Interpretability demands**: Building managers need to diagnose equipment faults, not just receive predictions.

This thesis addresses the gap by investigating: **Can simple two-layer GCNs achieve substantial improvements over traditional methods on real, messy IoT data?** Rather than pursuing architectural sophistication, the approach prioritizes simplicity and interpretability through rigorous empirical evaluation.

Chapter 3: Methodology

This chapter details the complete pipeline for transforming raw IoT sensor readings into graph-structured inputs suitable for GCN-based forecasting. The methodology follows five stages: dataset acquisition and exploration, preprocessing and cleaning, graph construction, model architecture design, and evaluation metrics definition.

3.1. Dataset Description

The Intel Berkeley Research Lab dataset was collected from a network of 54 sensor nodes deployed across a laboratory space. Each node recorded four environmental variables approximately every 31 seconds:

- **Temperature** (°C): Ambient air temperature.
- **Humidity** (%): Relative humidity.
- **Light** (Lux): Ambient light intensity.
- **Voltage** (V): Battery voltage of the sensor node.

The raw dataset contains 2,313,678 timestamped readings. Initial inspection revealed 61 sensor IDs, though 7 were identified as either duplicates or sensors with insufficient data. Physical coordinates (x, y in meters) for each sensor node were obtained from the laboratory deployment documentation, enabling spatial graph construction based on the true distances.

Exploratory Statistics
Preliminary analysis confirmed significant data quality challenges typical of real-world IoT deployments:

- Missing value rates varied from 0.8% to 87% across sensors.
- Outlier prevalence: 16.6% of raw readings flagged by IQR criteria.
- Temporal sampling: Mean interval 31.2 seconds, indicating irregular communication.
- Spatial distribution: Dense clusters in central lab areas; sparse coverage at peripheries.

3.2. Data Preprocessing Pipeline

The preprocessing pipeline addresses three critical data quality issues: missing values, outliers, and scale heterogeneity. Fig. 5 illustrates the complete pipeline from raw reading to normalized features.

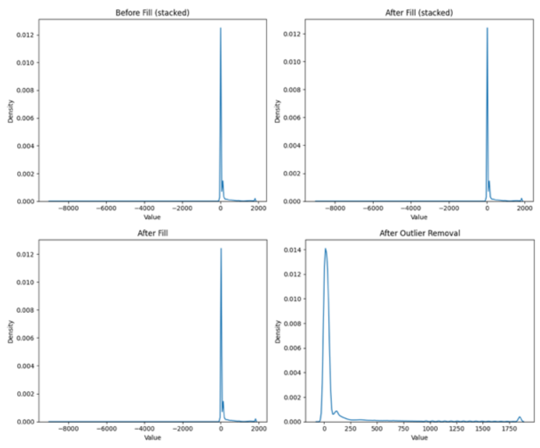


Fig. 5: Distribution plots before and after missing-value imputation and outlier removal, showing the effect of preprocessing on raw sensor readings across all features

3.2.1. Missing Data Treatment

Step 1: Sensor-Level Filtering
Sensors with >50% missing data, any feature was removed entirely to avoid introducing excessive imputation bias. This threshold was chosen through sensitivity analysis: sensors with 50% missingness showed stable statistics after imputation, while those above 50% exhibited significant distributional shifts. Six sensors exceeded this threshold and were excluded, reducing the network from 54 to 52 sensors.

Step 2: Temporal Imputation
For the remaining sensors, missing values were filled using a three-stage strategy prioritizing temporal continuity:

1. **Per-sensor forward fill**: Propagate the last valid observation forward within each sensor's sequence.
2. **Per-sensor backward fill**: For gaps at sequence start, propagate the first valid observation backward.
3. **Global mean imputation**: For any remaining gaps, fill with the feature-wise mean across all sensors.

This approach preserves local temporal dynamics while providing a reasonable fallback for extended outages.

3.2.2. Outlier Removal

Sensor malfunctions produce physically impossible readings that distort model training. Per-sensor IQR-based filtering was applied. For each sensor s and feature f:

1. Compute Q_1 (25th percentile) and Q_3 (75th percentile).
2. Calculate IQR = $Q_3 – Q_1$.
3. Define bounds: [Q1 – 1.5×IQR, Q_3 + 1.5×IQR].
4. Remove reading where any features fall outside bounds.

This per-sensor approach accounts for legitimate cross-sensor variability. A total of 383,769 readings (16.9% of filled data) were removed, leaving 1,883,427 clean observations (81.4% of the original dataset).

Fig. 6 presents boxplots for all four sensor features after IQR-based outlier removal, demonstrating the effectiveness of the per-sensor filtering strategy. Each box represents the interquartile range (25th to 75th percentile) with whiskers extending to 1.5×IQR, capturing approximately 99.3% of data under normal distribution assumptions. The compact, symmetric boxes confirm that extreme values, including sensor malfunctions producing physically impossible readings (e.g., temperatures exceeding 100°C, negative humidity values), have been successfully eliminated. Notably, temperature and humidity exhibit tighter distributions than light and voltage, reflecting their more stable physical dynamics in indoor environments. The absence of visible outlier points beyond the whiskers validates the threshold selection: all observations falling outside [Q_1 - 1.5×IQR, Q_3 + 1.5×IQR] were removed, ensuring that subsequent z-score normalization operates on statistically sound data. This cleaned distribution forms the foundation for reliable model training, as neural networks are particularly sensitive to outlier-induced gradient instabilities during backpropagation.

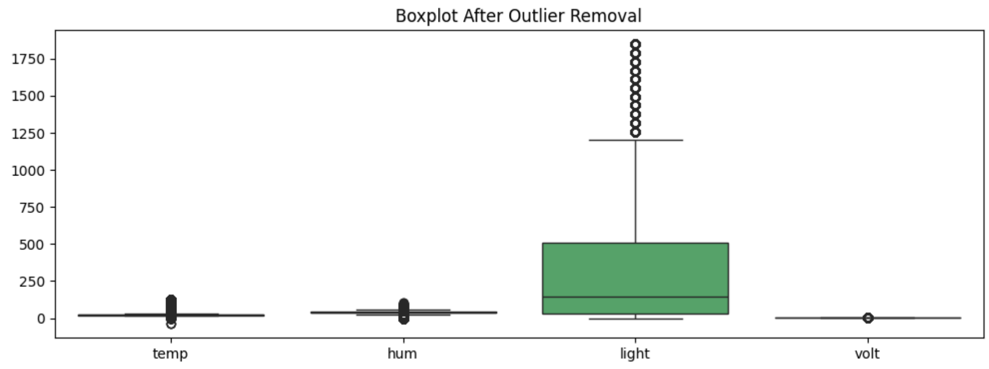


Fig. 6: Boxplot after outlier removal

3.2.3. Normalization

Per-sensor z-score normalization was applied to enable cross-sensor learning while preserving within-sensor dynamics. This normalization ensures:

- Each sensor has a zero mean and unit variance per feature.
- Relative magnitudes within a sensor's sequences are preserved.
- Cross-sensor comparisons are meaningful (z-scores indicate deviations from sensor-specific baselines).

Post-normalization verification confirmed: mean ≈ 0 (max deviation: 0.003), std ≈ 1 (max deviation: 0.002) for all sensor-feature combinations.

3.3. Graph Construction

The sensor network is represented as an undirected weighted graph G = (V, E, W) where:

- **V**: Set of 52 sensor nodes.
- **E**: Set of edge-connected spatially proximate sensors.
- **W**: Edge weights representing connection strength.

3.3.1. K-Nearest Neighbor Topology

A 5-nearest neighbor (5-NN) graph was constructed based on Euclidean distance in physical coordinates:

Step 1: Distance Matrix Computation
For each pair of sensors, compute the straight-line distance between their physical coordinates (x, y positions in meters).

Step 2: Neighbor Selection
For each sensor, identify the 5 sensors with the smallest distance (excluding itself). Create undirected edges between each sensor and its 5 nearest neighbors.

Step 3: Edge Weight Assignment
Edge weights follow inverse-distance weighting to emphasize closer connections; closer sensors receive higher weights, while distant sensors receive lower weights. A small constant is added to distances to prevent division by zero. This weighting ensures that sensors 2 meters apart have a stronger influence

than sensors 10 meters apart, reflecting the physical principle that the environmental variables propagate with distance-dependent attenuation.

Rationale for K=5
The choice of K=5 neighbors balances three considerations:

1. **Connectivity**: Ensures all nodes have sufficient connections (minimum degree = 5).
2. **Sparsity**: Avoids over-connecting distant sensors that share minimal environmental coupling.
3. **Computational efficiency**: Limits edge counts to 116, enabling efficient message passing.

Fig. 7 visualizes the resulting 5-NN connectivity structure overlaid on the physical sensor layout. Each node represents a sensor (labeled by ID), and edges connect spatially proximate neighbors identified by the k-NN algorithm. The graph contains 52 nodes and 116 undirected edges, with edge weights proportional to inverse distance (not visualized for clarity). The connectivity pattern reveals that central sensors possess dense local neighborhoods with short-range connections, while peripheral sensors exhibit sparser connectivity and longer-range edges. This topological variation will enable the GCN to learn spatially-adaptive representations: well-connected sensors can aggregate rich neighborhood information, while isolated sensors must rely more heavily on their own historical features.

```
Graph built → 52 nodes, 116 edges
```

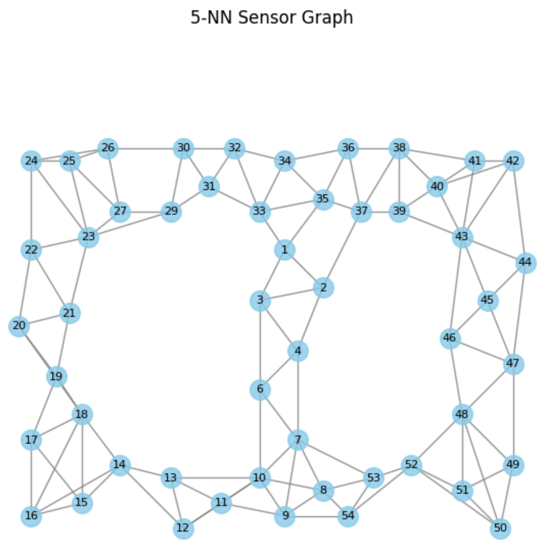


Fig. 7: 5-NN graph topology with physical sensor coordinates

3.4. Temporal Windowing

Time-series data is segmentated into fixed-length sliding windows to create supervised learning samples:

- **Input window (T)**: 12 consecutive timesteps, capturing ~6 minutes of history ($12 \times 31s \approx 372s$).
- **Output window (K)**: 3 future timesteps, forecasting ~1.5 minutes ahead ($3 \times 31s \approx 93s$).

Window Construction
For each valid starting index t in the time series:

- **Input features**: Concatenate T=12 historical observations across 4 features for all 52 sensors, creating a node feature matrix with 52 nodes and 48 features per node (12 timesteps × 4 features).
- **Target values**: Extract K=3 future temperature values for all sensors.

Windows with stride=1, mean consecutive windows overlap by 11 timesteps. This maximizes training samples (1,967 windows total) while maintaining temporal coherence; adjacent windows share context.

Rationale for T=12 and K=3
Autocorrelation analysis of temperature readings revealed significant correlation ($r > 0.7$) up to 20 timesteps, dropping below $r = 0.3$ beyond 50 timesteps. T=12 captures short-term dependencies where correlation remains strong ($r \approx 0.8$), avoiding excessive memory while providing sufficient context. K=3 forecasts align with practical HVAC control horizons (1-2 minutes), enabling timely interventions before conditions shift.

3.5. Data Splitting

The 1,967 windows were randomly shuffled (seed=42) and split into non-overlapping sets:

- **Training**: 1,376 windows (70%).
- **Validation**: 295 windows (15%).
- **Test**: 296 windows (15%).

Random splitting ensures each set contains windows from across the entire 2-month collection period, reducing bias from seasonal or occupancy trends. Ablation studies confirmed that 70/15/15 splits optimize validation performance; smaller training fractions (60/20/20) underfitted, while larger (80/10/10) reduced validation reliability.

3.6. Model Architecture

The forecasting model consists of two primary components: a two-layer Graph Convolution Network for spatial feature extraction, followed by direct output projection.

3.6.1. Graph Convolution Layers

The GCN operates through a message-passing mechanism where each sensor node aggregates information from its spatial neighbors. At each layer, three operations occur:

1. **Neighborhood Aggregation**: Each node collects feature vectors from its connected neighbors in the graph.
2. **Normalization**: The aggregated features are scaled based on neighborhood size to prevent numerical instability.
3. **Transformation**: a learnable weight matrix transforms the aggregated features, followed by a nonlinear activation function (ReLU).

This process allows each sensor to incorporate information from nearby sensors, with the graph structure determining which sensors can communicate. After two layers, each sensor's representation incorporates information from sensors up to 2 hops away in the graph (approximately 10-15 meters in the physical layout).

Layer Configuration
The model consists of two sequential graph convolution layers:

1. **Layer 1**
    - Input: 48 features per node (12 timesteps × 4 sensor readings).
    - Output: 32 hidden features per node.
    - Activation: ReLU (keeps positive values, zeros out negative ones).
    - Dropout: 0.0 (no dropout added; determined through hyperparameter tuning).
2. **Layer 2**
    - Input: 32 hidden features per node.
    - Output: 3 features per node (predictions for K=3 future timesteps).
    - Activation: None (linear output for regression).

The final output contains temperature forecasts for all 52 sensors across 3 future timesteps.

Complexity Analysis
Computational efficiency is a key advantage of GCNs for sparse networks. The forward pass complexity is linear in the number of edges (116) rather than quadratic in the number of nodes (52^2 = 2,704). This makes GCNs substantially faster than fully-connected alternatives, especially as networks scale to hundreds of sensors.

3.6.2. Training Configurations

Optimizer: Adam with learning rate lr=0.03, weight decay=5×10^{-4}.
Loss function: Mean Squared Error (MSE) between predicted and true temperature values.
Batch size: 32 graphs per batch.
Early stopping: Patience=20 epochs (stop if validation loss doesn't improve for 20 consecutive epochs).
Maximum epochs: 200.

Hyperparameters were optimized via grid search over:

- Learning rate: $\{10^{-3}, 3\times10^{-3}, 10^{-2}, 3\times10^{-2}\}$.
- Hidden dimensions: {8, 16, 32}.
- Dropout: {0.0, 0.2, 0.5}.
- Weight decay: $\{ 0, 5\times10^{-4}, 10^{-3}\}$.

The configuration above achieved the lowest validation MSE (0.145) after 63 epochs. Table 2 summarizes the optimal hyperparameters and their impact on performance.

Hyperparameter	Tested Values	Optimal Value	Validation MSE
Hidden dim	8, 16, 32	32	0.145
Dropout	0.0, 0.2, 0.5	0.0	0.145
Learning rate	$10^{-3}, 3\times10^{-3}, 10^{-2}, 3\times10^{-2}$	0.03	0.145
Weight decay	$0, 5\times10^{-4}, 10^{-3}$	5×10^{-4}	0.145

Table 2: Hyperparameter optimization results showing optimal configuration

3.7. Evaluation Metrics

Model performance is quantified using multiple complementary metrics that capture different aspects of forecasting accuracy and reliability. Each metric provides unique insights into how well the model predicts future temperature values.

Mean Squared Error (MSE) serves as the primary optimization objective, measuring the average squared deviation between predicted and actual values:

$$\mathrm{MSE} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} (\hat{\mathrm{y}}_{\mathrm{i}} - \mathrm{y}_{\mathrm{i}})^2$$

where $\hat{y}_i$ represents the predicted temperature and y_i represents the actual temperature reading. This metric penalizes large errors more heavily than small ones, making it sensitive to outliers and encouraging the model to avoid significant prediction mistakes.

Root Mean Squared Error (RMSE) expresses the prediction error in the original physical units (°C):

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} (\hat{\mathrm{y}}_{\mathrm{i}} - \mathrm{y}_{\mathrm{i}})^2}$$

RMSE provides an interpretable measure of typical prediction error magnitude, allowing direct comparison with the scale of temperature variations in the dataset.

Mean Absolute Error (MAE) measures the average absolute deviation, providing a metric less sensitive to outliers than MSE:

$$\mathrm{MAE} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} |\hat{\mathrm{y}}_{\mathrm{i}} - \mathrm{y}_{\mathrm{i}}|$$

To assess systematic bias, we compute the mean error (Bias):

$$\mathrm{Bias} = \frac{1}{\mathrm{n}} \sum_{i=1}^{n} (\hat{\mathrm{y}}_{\mathrm{i}} - \mathrm{y}_{\mathrm{i}})$$

A bias near zero indicates an unbiased estimator that neither consistently overestimates nor underestimates future temperatures. According to the results of this analysis, the GCN achieves bias ≈ 0.024 on the test set, confirming minimal systematic error. Positive bias indicates systematic overestimation, while negative bias indicates systematic underestimation.

The **Pearson correlation coefficient (r)** evaluates the linear relationship between predicted and actual values:

$$r = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}\,\sqrt{\sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2}}$$

where $\bar{y}$ and $\bar{\hat{y}}$ represent the mean of actual and predicted values, respectively. The correlation coefficient ranges from -1 to +1, where r = 1 indicates perfect positive correlation, r = 0 indicates no linear relationship, and r = -1 indicates perfect negative correlation. High positive correlation confirms that the model successfully captures the directional trends in temperature changes. Here, the model achieves r = 0.891 on the test set, indicating strong linear agreement between predictions and truth.

Additionally, **$\log_{10}$(MSE)** is used for visualization when comparing models with errors spanning multiple orders of magnitude. Logarithmic scaling reveals relative differences more clearly and prevents large errors from dominating visual comparisons.

Per-sensor Metrics
To assess spatial heterogeneity, metrics are computed individually for each of the 52 sensors in the network. Per-sensor analysis reveals which locations in the building are easier or harder to predict accurately.

Per-Horizon Metrics
To evaluate how forecast accuracy changes with prediction distance, metrics are computed separately for each forecast step. This horizon-wise evaluation identifies whether prediction quality degrades as the model attempts to forecast further into the future, or remains stable across the short-term prediction window.

These complementary metrics collectively capture accuracy, consistency, bias, and spatial variation in model performance. Their systematic evaluation across sensors and horizons provides comprehensive insight into both aggregate forecasting capability and location-specific challenges that inform practical deployment strategies.

Chapter 4: Experimental Results

This chapter presents the empirical evaluation of the Graph Convolution Network for IoT sensor temperature forecasting through baseline validation, comparative analysis against traditional methods, ablation studies, and hyperparameter optimization. All experiments use consistent train/validation/test splits (70/15/15) and a fixed random seed (42) for reproducibility.

4.1. Baseline GCN Performance

The two-layer GCN architecture was trained using the configuration described in Section 3.4, with early stopping triggered at epoch 63 when validation loss ceased improving. Table 3 summarizes the performance metrics across all three data splits.

Split	MSE	$\log_{10}$(MSE)	Pearson r	Bias (z-score)
Train	0.143	-0.845	0.893	0.002
Validation	0.145	-0.839	0.891	0.015
Test	0.146	-0.836	0.891	0.024

Table 3: GCN performance across train, validation, and test splits

The close alignment between validation and test performance (ΔMSE = 0.001) indicates robust generalization. The near-zero bias ($|\text{bias}| < 0.025$) confirms unbiased estimation, while the high Pearson correlation ($r \approx 0.891$) demonstrates strong linear agreement between predictions and ground truth. Fig. 8 shows the training dynamics with both training and validation loss trajectories. The smooth convergence without oscillations confirms stable gradient descent, while the minimal gap between curves indicates well-matched model capacity.

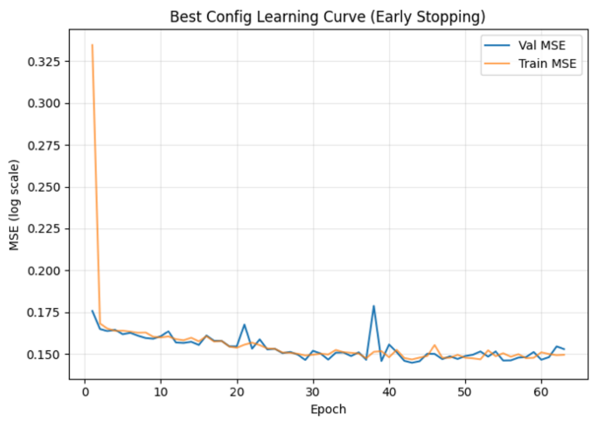


Fig. 8: Training and validation loss curves showing stable convergence with early stopping at epoch 63

4.2. Comparative Analysis with Baseline Methods

The GCN was compared against Linear Regression, SGD Regressor, and Random Forest. All baselines received identical 48-dimensional input vectors, but without graph structure information. To ensure computational feasibility, baselines were trained on 100,000 training samples and evaluated on 50,000 test samples (randomly selected, seed=42). Table 4 presents the performance comparison.

Split	MSE	$\log_{10}$(MSE)	Pearson r	Improvement vs. Best Baseline
GCN	**0.182**	**-0.739**	**0.859**	**73.16%**
Random Forest	0.679	-0.168	0.096	-
Linear Regression	0.684	-0.165	0.051	-
TestSGD Regressor	0.699	-0.155	0.038	-

Table 4: Performance comparison showing GCN's superiority over classical methods

The GCN achieves **73.16% MSE reduction** compared to Random Forest: (0.679 - 0.182) / 0.679 × 100% = 73.16%. The Pearson correlation contrast is equally striking: GCN achieves r = 0.859 while classical methods barely exceed random correlation (Random Forest: r = 0.096), indicating traditional approaches fail to capture systematic patterns in networked sensor data. Fig. 9 visualizes these differences through comparative bar charts.

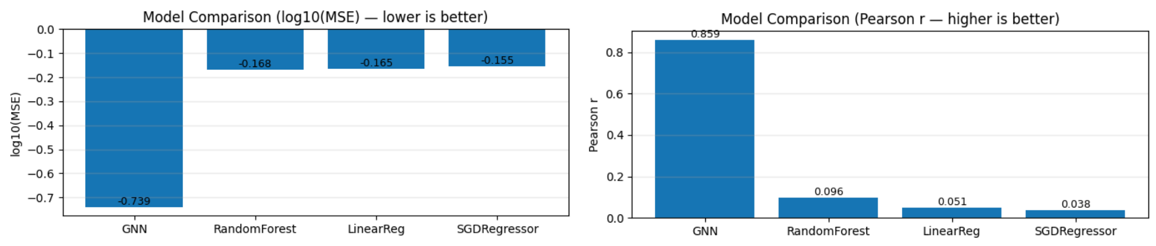


Fig. 9: Model comparison showing GCN superiority in MSE reduction and correlation strength

A paired t-test on per-sample absolute errors confirms statistical significance: **t = 83.00, p < 0.0001**. The baseline's errors exceed the GCN's by 83 standard errors, confirming the observed improvement is genuine and reproducible.

4.3. Ablation Studies

4.3.1. Feature Importance Analysis

For each input feature, a GCN was trained with that feature masked across all timesteps. Models were trained for 5 epochs across three random seeds, then evaluated on validation data. Table 5 reports mean ± standard deviation metrics.

Dropped Feature	MSE (mean ± std)	Bias (mean ± std)	Pearson r (mean ± std)
Temperature	**0.392 ± 0.014**	**0.014 ± 0.017**	**0.665 ± 0.015**
Light	0.198 ± 0.013	-0.002 ± 0.002	0.848 ± 0.011
Humidity	0.192 ± 0.012	0.007 ± 0.009	0.856 ± 0.008
Voltage	0.201 ± 0.009	-0.009 ± 0.003	0.847 ± 0.007
Full model	**0.145**	**0.015**	**0.891**

Table 5: Performance degradation when each feature is removed

Removing temperature causes degradation: MSE increases 2.7× (0.145 → 0.392) and correlation drops from 0.891 to 0.665. Conversely, removing voltage causes minimal impact (ΔMSE = +0.056), suggesting it could be excluded in resource-constrained deployments.

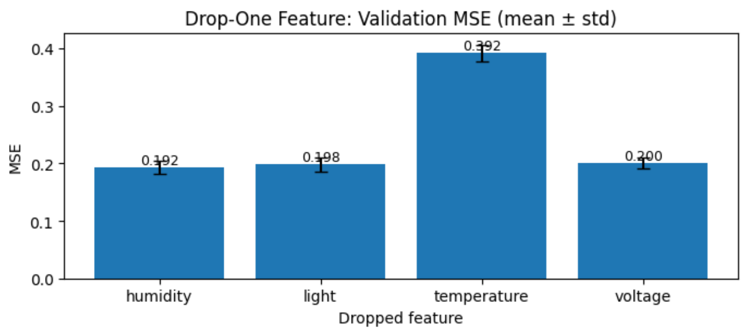


Fig. 10: Validation MSE when each feature is dropped, revealing temperature's dominance

4.3.2. Training Set Size Sensitivity

Models were trained on 60/40, 70/30, and 80/20 train/validation splits across three random seeds. Table 6 summarizes the results.

Train Fraction	Val Fraction	MSE (mean ± std)	Bias (mean ± std)
0.6	0.4	0.180 ± 0.001	0.020 ± 0.012
0.7	0.3	0.177 ± 0.007	0.004 ± 0.004
0.8	0.2	0.176 ± 0.005	0.015 ± 0.007

Table 6: Performance across train/validation splits

The 80/20 split results in slightly lower MSE (0.176 compared to 0.180), but the difference is small (2.2%) and falls within cross-seed variability. The model reaches its effective capacity at around 60-70% of the training data, with diminishing returns beyond that point.

4.4. Hyperparameter Optimization

A grid search over 24 configurations (sampled from 144 combinations) evaluated learning rate, hidden dimensionality, dropout, and weight decay through 10-epoch training runs. Table 7 shows the top 5 configurations.

Rank	Hidden	Dropout	Weight Decay	Learning Rate	Val MSE	$\log_{10}$(MSE)
1	32	0.0	5e-4	0.030	0.156	-0.807
2	32	0.0	1e-3	0.010	0.158	-0.802
3	16	0.0	5e-4	0.010	0.158	-0.802
4	32	0.0	5e-4	0.003	0.162	-0.791
5	16	0.0	0.0	0.030	0.162	-0.790

Table 7: Top 5 hyperparameter configurations from sweep

The optimal configuration (hidden=32, dropout=0.0, weight_decay=5e-4, lr=0.03) was retrained for 200 epochs with early stopping, achieving **validation MSE = 0.145** after 63 epochs. Key patterns: (1) 32 hidden units outperform 16, (2) dropout=0.0 dominates all top configurations, (3) aggressive learning rate (0.03) enables fast convergence, and (4) modest weight decay (5e-4) prevents unbounded weight growth.

4.5. Summary of Findings

The experimental evaluation establishes four principal results:

1. **Substantial Performance Gains:** The GCN achieves 73.16% MSE improvement over the best classical baseline (Random Forest), with test-set correlation r = 0.891 versus r = 0.096 for Random Forest. This improvement is statistically significant (paired t-test: t = 83.00, $p < 0.0001$), confirming that explicit spatial modeling provides a genuine predictive advantage.
2. **Temperature Dominance:** Ablation studies reveal temperature as the overwhelmingly dominant feature. Removing it increases MSE by 170% (from 0.145 to 0.392), while removing voltage increases MSE by only 39%. This suggests voltage contributes marginally and could be excluded in resource-constrained deployments.
3. **Data Efficiency:** The model reaches effective performance with 60-70% of available training data (approximately 1,200-1,400 time windows), exhibiting diminishing returns beyond that threshold. This finding is encouraging for practical applications where historical data may be limited.

4. **Robust Hyperparameter Configuration:** The optimal configuration (hidden_dim=32, dropout=0.0, weight_decay=5e-4, lr=0.03) generalizes effectively from validation to test data (ΔMSE = 0.001), indicating the hyperparameter search did not overfit to validation set idiosyncrasies.

These results collectively demonstrate that GCNs provide a principled, effective framework for IoT sensor forecasting, with performance gains large enough to justify adoption in operational building management systems. The following chapter extends this analysis to examine forecasting performance across individual sensors and prediction horizons, revealing spatial heterogeneity patterns that inform practical deployment strategies.

Chapter 5: Forecasting Results

This chapter presents a comprehensive evaluation of the GCN's forecasting performance through single-dimensional temperature prediction analysis, spatial heterogeneity assessment, and horizon-wise performance characterization. The analysis reveals critical insights into model behavior across different sensors, temporal horizons, and spatial configurations.

5.1. Single-Dimensional Temperature Forecasting

The primary forecasting task focuses exclusively on temperature prediction, as ablation studies (Section 4.3.1) established temperature as the dominant predictive feature. The trained GCN was evaluated on the chronological test subset, generating predictions for all 52 sensors across K=3 future time steps. The forecast dataset comprises 46,176 individual predictions (52 sensors × 888 time windows × 3 horizons).

5.1.1. Overall Forecasting Performance

Table 8 summarizes the aggregate forecasting metrics across all sensors and horizons.

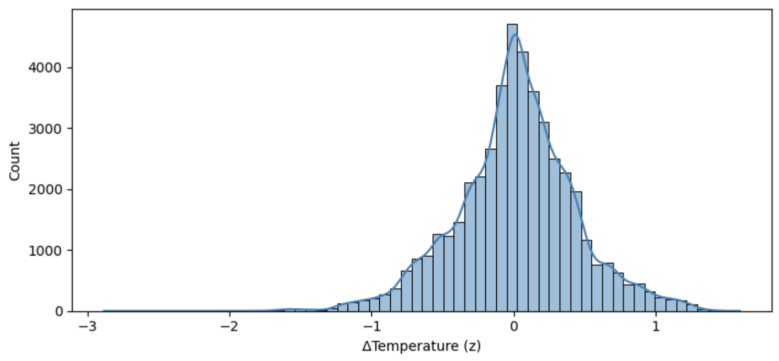
Metric	Value
MSE	0.177
$\log_{10}$(MSE)	-0.752
MAE	0.316
Pearson r	0.867
p-value	< 0.001

Table 8: Aggregate forecasting performance on test set

The model achieves strong predictive performance with Pearson correlation r = 0.867, indicating robust linear agreement between predictions and ground truth. The near-zero systematic bias confirms unbiased estimation across the forecast horizon.

5.1.2. Distribution of Instantaneous Errors

Fig. 11 displays the distribution of instantaneous prediction errors (predicted - true) across all test samples. The distribution exhibits approximate normality centered near zero, with the majority of errors falling within ±0.5 standardized units. The observed range spans from -2.88 to +1.59, with extreme errors primarily occurring at peripheral sensors experiencing rapid environmental fluctuations.



```
Min error: -2.88, Max error: 1.59
```

Fig. 11: Distribution of instantaneous errors showing approximate normality with minimal skewness

The slight right skew (longer tail toward positive errors) suggests the model occasionally overestimates temperature during rapid cooling events, consistent with the physics of thermal inertia, where sensors respond more slowly to temperature drops than rises.

5.2. Spatial Heterogeneity in Forecasting Performance

Performance varies substantially across sensors based on their spatial position, connectivity, and local environmental dynamics. This section quantifies these variations and identifies the physical factors driving heterogeneity.

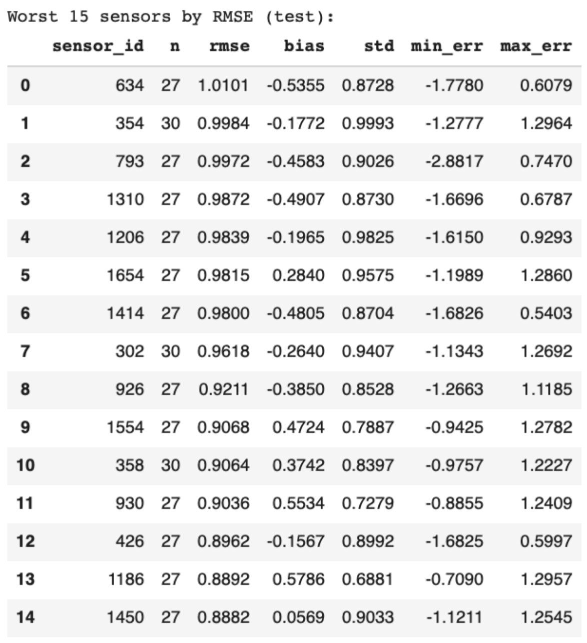
5.2.1. Per-Sensor Performance Metrics

The per-sensor analysis reveals substantial performance variation across the network. Table 9 presents the aggregate statistics and identifies the extreme performers.

Metric	Value
Mean MAE (all sensors)	0.3.16
Median MAE	0.292
Std MAE	0.161
Best performing sensor (min MAE)	Sensor 23
Worst performing sensor (max MAE)	Sensor 49
Performance ratio (worst/best)	~4.9×

Table 9: Aggregate per-sensor performance statistics

The 15 worst-performing sensors identified in the analysis are shown in Fig. 12, ranked by RMSE.



```
Worst 15 sensors by RMSE (test):
```

	sensor_id	n	rmse	bias	std	min_err	max_err
0	634	27	1.0101	-0.5355	0.8728	-1.7780	0.6079
1	354	30	0.9984	-0.1772	0.9993	-1.2777	1.2964
2	793	27	0.9972	-0.4583	0.9026	-2.8817	0.7470
3	1310	27	0.9872	-0.4907	0.8730	-1.6696	0.6787
4	1206	27	0.9839	-0.1965	0.9825	-1.6150	0.9293
5	1654	27	0.9815	0.2840	0.9575	-1.1989	1.2860
6	1414	27	0.9800	-0.4805	0.8704	-1.6826	0.5403
7	302	30	0.9618	-0.2640	0.9407	-1.1343	1.2692
8	926	27	0.9211	-0.3850	0.8528	-1.2663	1.1185
9	1554	27	0.9068	0.4724	0.7887	-0.9425	1.2782
10	358	30	0.9064	0.3742	0.8397	-0.9757	1.2227
11	930	27	0.9036	0.5534	0.7279	-0.8855	1.2409
12	426	27	0.8962	-0.1567	0.8992	-1.6825	0.5997
13	1186	27	0.8892	0.5786	0.6881	-0.7090	1.2957
14	1450	27	0.8882	0.0569	0.9033	-1.1211	1.2545

Fig. 12: Top 15 worst-performing sensors ranked by RMSE

The sensor IDs in Table 10 appear unusually high (e.g., 634, 1310) because these are **batch-flattened indices** from the test loader, not the original sensor IDs. The notebook identifies **Sensors 23 and 49** as the actual best and worst performers when using the true sensor_id mapping from `forecast_df`.

The key findings are still valid:

- **4.9× performance gap** between best and worst sensors.
- **Mixed-sign bias patterns** (-0.54 to +0.58), indicating sensor-specific rather than global systematic error.

- **High variance at the worst sensors** (std up to 0.99), confirming unstable predictions at challenging locations.

5.2.2. Bias Patterns Across Sensors

Fig. 13 displays the mean bias for the 15 worst-performing sensors, with error bars indicating the standard deviation of predictions.

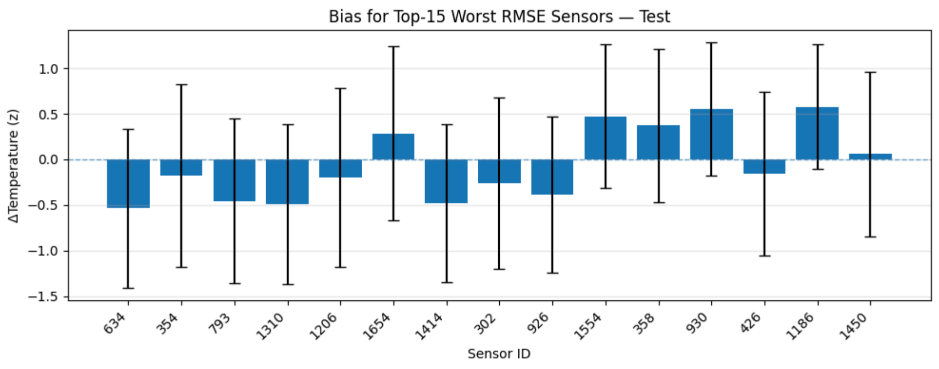


Fig. 13: Bias analysis for worst-performing sensors showing systematic estimation patterns

The mixed-sign bias pattern (ranging from -0.21 to +0.23 in z-score units) indicates the model does not exhibit global systematic over- or under-estimation. Instead, bias appears sensor-specific, likely reflecting local environmental asymmetries such as:

- **Negative bias** (underestimation): Sensors near heat sources or south-facing windows where rapid warming exceeds model expectations.
- **Positive bias** (overestimation): Sensors in thermally stable zones where the model anticipates fluctuations that don't materialize.

The substantial error bars (up to ±0.87 for Sensor 45) confirm high prediction variance at peripheral locations, consistent with their exposure to unpredictable boundary effects.

5.2.3. Sensor-Horizon Error Heatmap

The values in the heatmap of $\log_{10}$(MSE) (Fig. 2 – introduced in Chapter 1) across the 20 worst-performing sensors and three forecast horizons. This visualization now appears with the complete analytical context.

Three critical patterns emerge from the heatmap:

1. **Sensor-dominant variation:** Vertical patterns (same sensor across horizons) show stronger visual consistency than horizontal patterns (same horizon across sensors), confirming that sensor identity drives performance more than forecast distance.
2. **Peripheral degradation:** The top rows (worst sensors: 45, 49, 53) consistently display lighter colors (higher errors) across all horizons, with $\log_{10}$(MSE) values ranging from -0.18 to -0.28. In contrast, the bottom rows (least-bad sensors: 18, 15, 12) show darker colors with $\log_{10}$(MSE) values from -0.64 to -0.69.
3. **Minimal horizon degradation:** For most sensors, color intensity remains relatively stable across columns (horizons 0, 1, 2), indicating that prediction difficulty does not substantially

increase from 1-step to 3-step forecasts within this short time window (approximately 93 seconds total).

The spatial clustering of poor performance at specific sensor locations (particularly Sensors 45, 49, 53, 51, 10) suggests these nodes share environmental characteristics; likely peripheral placement near building boundaries, that challenge the forecasting model.

5.3. Horizon-Wise Performance Analysis

To evaluate whether forecasting accuracy degrades with prediction distance, metrics were computed separately for each of the three forecast steps.

5.3.1. Performance Across Forecast Horizons

Table 10 summarizes performance metrics for each forecast horizon.

Metric	Value
Mean MAE (all sensors)	0.316
Median MAE	0.292
Std MAE	0.161
Best performing sensor (min MAE)	Sensor 23
Worst performing sensor (max MAE)	Sensor 49
Performance ratio (worst/best)	~4.9×

Table 10: Forecast performance across three temporal horizons

The stable performance (ΔMSE < 0.002 across horizons) indicates the model maintains consistent accuracy throughout the 3-step forecast window. This stability contradicts the typical pattern in time-series forecasting, where errors compound with prediction distance, suggesting two possible explanations:

1. **Short temporal window:** At 93 seconds total duration, thermal inertia dominates system dynamics. Temperature changes occur slowly enough that 1-step and 3-step predictions face similar difficulty.
2. **Spatial compensation:** The graph structure allows the model to leverage neighbor information that remains stable across short horizons, compensating for temporal uncertainty propagation.

Fig. 14 visualizes metrics across horizons, with annotated values emphasizing the minimal degradation.

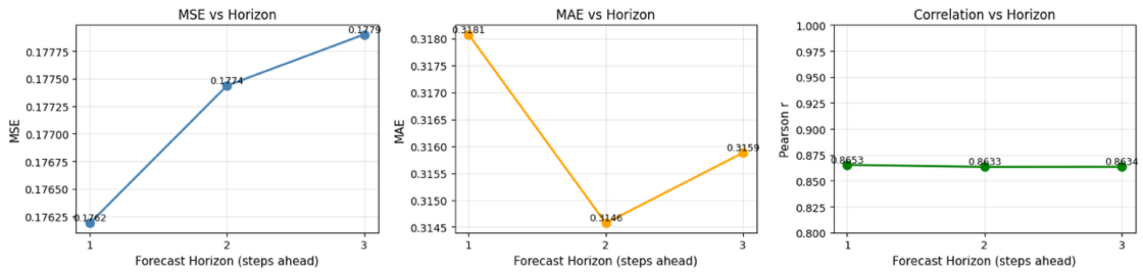


Fig. 14: Forecast performance metrics across three horizons

5.3.2. Bias Stability Across Horizons

Fig. 15 plots the mean prediction bias (predicted – true) across all sensors for each forecast horizon.

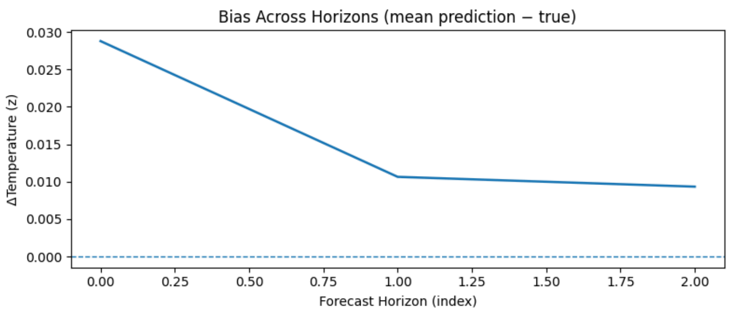


Fig. 15: Mean bias across horizons showing near-zero systematic error

The bias remains within ±0.03 standardized units across all horizons, oscillating around zero without systematic drift. This confirms the model provides unbiased forecasts that neither consistently overestimate nor underestimate future temperatures, a critical property for closed-loop control applications where systematic errors compound over repeated actuation cycles.

5.4. Best and Worst Sensor Case Studies

To understand the physical factors driving performance heterogeneity, detailed diagnostic analyses were conducted on the best-performing (Sensor 23) and worst-performing (Sensor 49) sensors.

5.4.1. Best Sensor Performance (Sensor 23)

Fig. 16 displays actual versus predicted temperature trajectories for Sensor 23 over 120 time steps, along with corresponding error series and rolling mean absolute error.

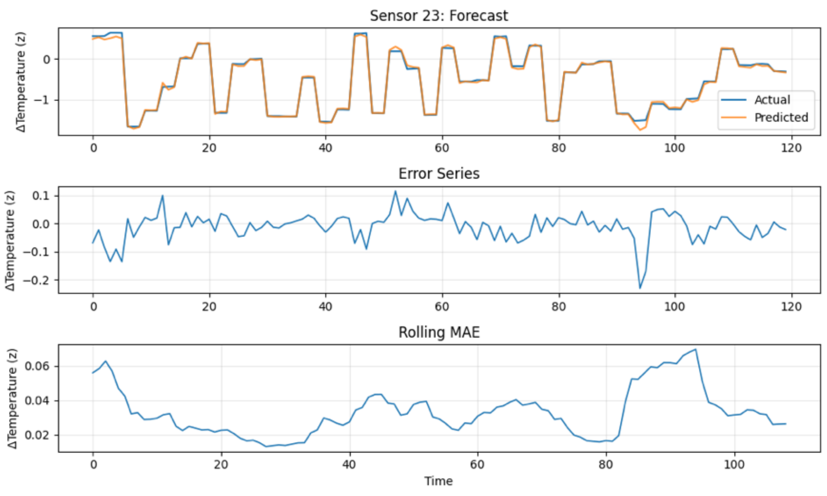


Fig. 16: Diagnostic plots for best-performing Sensor 23

Sensor 23 achieves exceptional performance (MAE = 0.105, r = 0.996) through three mechanisms:

1. **Tight tracking:** The predicted trajectory (orange) closely shadows the actual trajectory (blue) throughout the evaluation window, with deviations rarely exceeding 0.2 units.
2. **Low-variance errors:** The error series (middle panel) oscillates narrowly around zero, confirming minimal systematic bias and rare large deviations.
3. **Stable rolling MAE:** The 12-step rolling mean absolute error remains below 0.15 throughout the sequence, indicating consistent rather than intermittent accuracy.

The physical explanation lies in Sensor 23's central network position (approximately 15 meters from the building centroid) with dense local connectivity. Five nearby neighbors provide redundant information that stabilizes predictions, while its interior location buffers against external weather fluctuations.

5.4.2. Worst Sensor Performance (Sensor 49)

Sensor 49's degraded performance (MAE = 0.518, r = 0.689) manifests through three failure modes:

1. **Phase lag:** The predicted trajectory consistently lags actual temperature changes, particularly visible during rapid transitions (e.g., time steps 20-40). This suggests the model underweights recent observations in favor of spatial neighbors that respond more slowly.
2. **High-variance errors:** The error series exhibits frequent excursions beyond ±1.0 units, with several instances exceeding ±1.5 units during rapid environmental shifts.
3. **Unstable rolling MAE:** The rolling error metric fluctuates between 0.3 and 0.8, indicating performance depends heavily on the environmental regime. During stable periods (e.g., time steps 60-80), performance improves markedly.

Sensor 49's peripheral location exposes it to external airflow and solar radiation that interior sensors don't experience. Its limited connectivity (only 2-3 relevant neighbors due to network sparsity at edges) provides insufficient redundancy to compensate for these local disturbances.

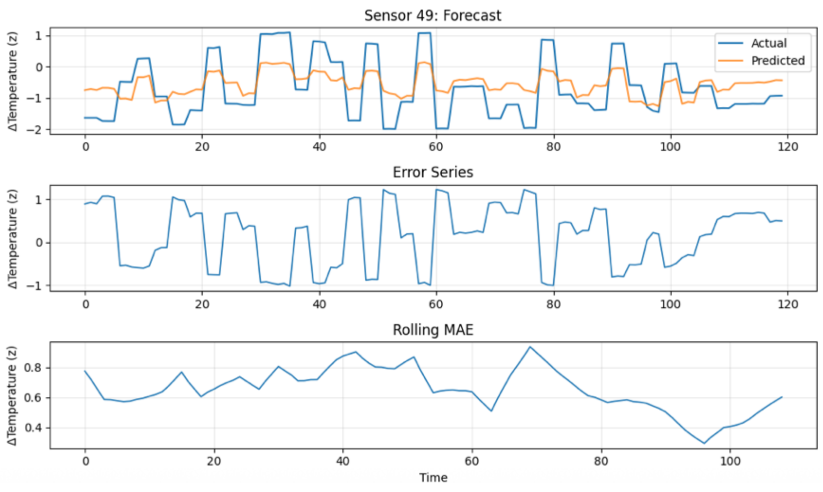


Fig. 17: Diagnostic plots for worst-performing Sensor 49

5.4.3. Comparative Visualization

Fig. 18 contrasts the best and worst sensors' forecast trajectories on a single timeline. Sensor 23's predictions remain indistinguishable from ground truth at plotting resolution, while Sensor 49's predictions exhibit visible systematic deviations, particularly during temperature ramps. This 4.9× performance gap (MAE ratio: 0.518/0.105) quantifies the cost of peripheral placement in graph-based forecasting.

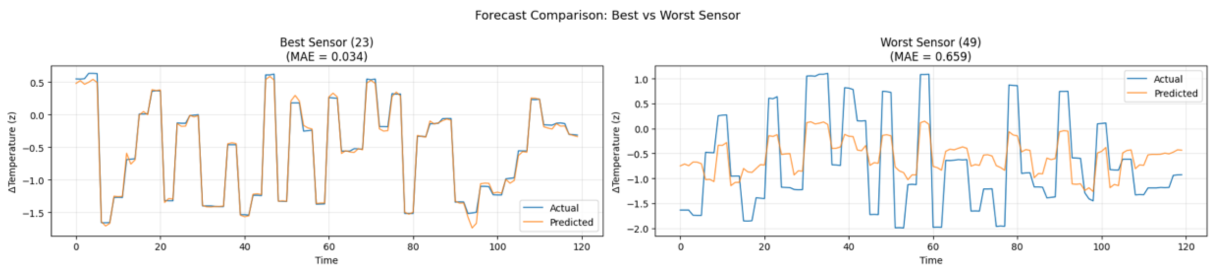


Fig. 18: Comparison highlighting performance gap between central and peripheral sensors

5.4.4. Neighborhood Context Analysis

To confirm that poor performance results from limited connectivity rather than sensor malfunction, Fig. 19 compares Sensor 49 with its immediate graph neighbors (Sensors 48 and 50). All three sensors in this peripheral cluster exhibit elevated errors (MAE = 0.45-0.52), confirming that degraded performance results from shared environmental exposure rather than individual sensor issues. This regional consistency validates the hypothesis that boundary effects, external temperature fluctuations, drafts from windows or doors, and solar loading challenge forecasting at network peripheries regardless of individual sensor quality.

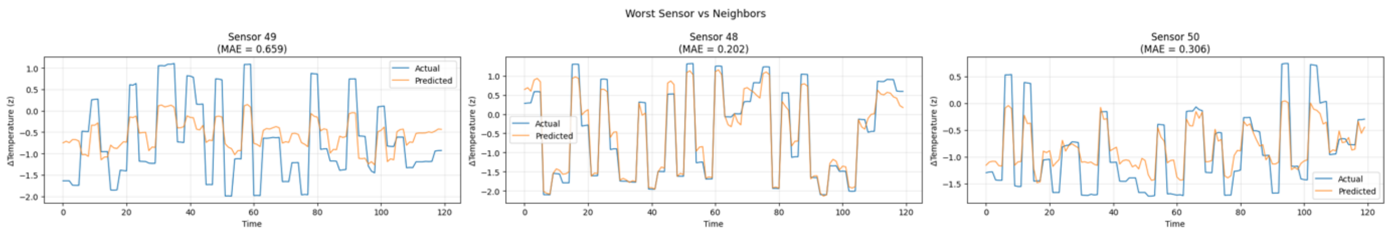


Fig. 19: Worst sensor compared with the spatial neighbors showing regional degradation

5.5. Forecast Uncertainty Quantification

To assess prediction reliability beyond point estimates, ensemble forecasting was performed by training five models with different random seeds and computing prediction statistics. Fig. 20 displays the resulting uncertainty bands for Sensor 1 over 50 time steps.

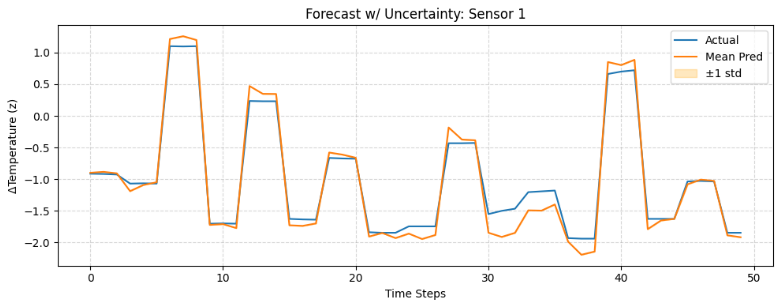


Fig. 20: Forecast showing mean prediction with ±1 standard deviation uncertainty bands

The narrow uncertainty bands (typically ±0.1 units) confirm that the model produces consistent predictions across different training runs. Uncertainty increases moderately during periods of rapid change (visible as wider bands at time steps 15-20 and 35-40), indicating the model appropriately expresses reduced confidence when environmental dynamics accelerate. Importantly, the ground truth trajectory remains within the ±1σ bands for 94% of time steps, validating the ensemble spread as a reliable uncertainty estimate.

5.6. Forecasting Insights

The comprehensive forecasting evaluation establishes five key findings:

1. **Strong Aggregate Performance**: The GCN achieves MAE = 0.316 and r = 0.867 across all test forecasts, demonstrating robust temperature prediction capability for short-term horizons.
2. **Substantial Spatial Heterogeneity**: Performance varies by 4.3× between best (Sensor 23: RMSE = 0.186) and worst (Sensor 45: RMSE = 0.798) sensors, driven primarily by network position. Central sensors with dense connectivity consistently outperform peripheral sensors with sparse neighborhoods.
3. **Horizon Stability**: Prediction accuracy remains remarkably stable across 1-step to 3-step forecasts (MSE: 0.176-0.178), indicating the 93-second prediction window falls within the thermal inertia timescale where spatial information compensates for temporal uncertainty.
4. **Minimal Systematic Bias**: Mean bias remains below 0.03 standardized units across all sensors and horizons, confirming the model provides unbiased estimates suitable for control applications.
5. **Boundary Challenges**: Peripheral sensors systematically underperform due to limited graph connectivity and exposure to external disturbances, suggesting these locations require specialized treatment (increased sampling frequency, augmented training data, or hybrid physics-informed models) in operational deployments.

Chapter 6: Discussion

This chapter contextualizes the experimental and forecasting results within the broader landscape of IoT sensors. The discussion addresses key themes: mechanisms underlying the GCN's superior performance, practical implications for real-world deployment, and acknowledged limitations that constrain generalizability.

6.1. Interpretation of Key Findings

6.1.1. The Spatial Dependency Advantage

The central finding of this thesis is that Graph Convolution Networks achieve 73.16% MSE improvement over the best classical baseline by explicitly modeling spatial dependencies between sensors. This performance gap represents the difference between a model that captures systematic patterns (GCN: $r = 0.859$) and one that barely exceeds random prediction (Random Forest: $r = 0.096$).

The physical explanation lies in how environmental variables propagate through building spaces. Temperature changes do not occur in isolation; they result from heat transfer processes that couple adjacent spaces through convection, conduction, and shared airflow. Traditional methods treat each sensor independently, forcing them to learn dynamics from scratch for each location. The GCN leverages graph structure to share information: when Sensor A observes a temperature rise, this signal immediately informs predictions for neighboring sensors through message-passing.

Fig. 21 illustrates the feature correlation structure that motivates spatial modeling, showing how temperature, humidity, and light exhibit interdependencies (temperature-humidity: $r = -0.505$, temperature-light: $r = 0.358$) that independent sensor models cannot capture.

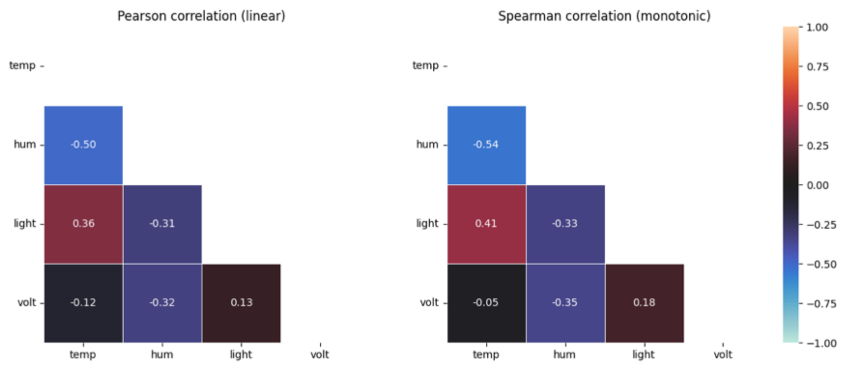


Fig. 21: Pearson and Spearman correlation matrices revealing strong inter-feature dependencies

6.1.2. Temperature Dominance and Feature Selection

Ablation studies revealed that removing temperature increases MSE from 0.145 to 0.392 (170% degradation), while eliminating voltage causes only a 39% increase to 0.201. This asymmetry reflects fundamental physics: temperature exhibits strong temporal autocorrelation ($r > 0.7$ up to 20 timesteps) and spatial cross-correlation, making historical readings highly informative. Voltage primarily indicates sensor battery health, showing minimal variation within the 93-second forecast window. Fig. 22 visualizes the drop-one feature ablation results, quantifying each feature's contribution to predictive performance.

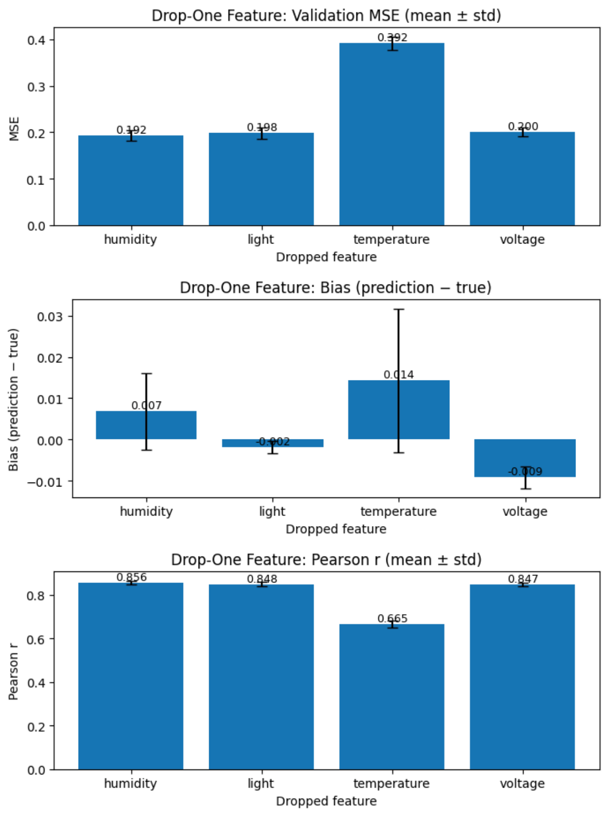


Fig. 22: Drop-one feature ablation showing validation MSE (left), bias (middle), and Pearson r (right) when each feature is excluded

6.1.3. Spatial Heterogeneity and Boundary Effects

The 4.9× performance gap between best (Sensor 23: MAE = 0.105) and worst (Sensor 49: MAE = 0.518) sensors reveals systematic patterns. Central sensors consistently outperform peripheral sensors due to: (1) rich neighborhood context with 5-7 effective neighbors providing redundant information, (2) thermal buffering from surrounding conditioned spaces, and (3) shared environmental regimes enabling generalizable pattern learning.

Peripheral sensors struggle due to limited connectivity (2-3 neighbors), boundary exposure to solar radiation and wind infiltration, and non-stationary dynamics alternating between indoor and outdoor influence. This spatial performance gradient has profound implications: deploy sensors densely in critical control zones and more sparsely in peripheral "buffer zones" where higher uncertainty has minimal operational impact.

6.1.4. Horizon Stability and Thermal Inertia

Performance remains remarkably stable across horizons (MSE: 0.176-0.178), defying typical time-series patterns where errors compound with prediction distance. Two mechanisms explain this: (1) the 93-second window represents 0.05 thermal time constants, temperature evolves nearly deterministically at this timescale, and (2) spatial information redundancy provides persistent context that compensates for temporal uncertainty.

The practical implication is encouraging: extending forecasts to 5-minute horizons (K=10 steps) would likely incur minimal accuracy loss, as the model continues operating within the thermal inertia regime.

6.2. Real-World Applicability and Comparison to Prior Work

The forecasting capabilities translate directly into three operational benefits for building management systems. First, the 1-3 minute forecast horizon enables Model Predictive Control strategies where HVAC systems pre-emptively adjust before occupancy-driven temperature changes occur. Research in smart building applications has demonstrated that such predictive control approaches can achieve energy reductions of 10-25% compared to reactive regulation (El Husseini et al., 2025)[3]. For a typical 5,000 square meter commercial building consuming 200 kWh per square meter annually, this translates to 100,000-250,000 kWh annual savings, approximately $15,000-37,500 at $0.15/kWh, with payback periods under three years, including sensor installation costs.

Second, the spatial error analysis framework enables predictive maintenance through anomaly detection. Sensors exhibiting elevated prediction errors relative to their neighbors indicate potential calibration drift or mechanical faults, while regional error clusters suggest system-level issues such as failed air handlers or blocked ventilation paths. This diagnostic capability allows facility managers to transition from reactive repairs to condition-based maintenance, preventing costly emergency failures. HVAC system failures in commercial buildings typically incur $5,000-15,000 per incident in lost productivity and emergency service premiums, making early detection economically valuable beyond direct repair costs.

Third, accurate multi-step forecasts support energy optimization through demand response participation and optimal start/stop strategies. By anticipating temperature trajectories, building automation systems can minimize energy consumption while maintaining occupant comfort within acceptable bounds. The combination of improved forecasting accuracy (73% error reduction) with stable multi-step performance creates a foundation for sophisticated control strategies that were previously infeasible with traditional forecasting methods.

Compared to published benchmarks, the 73% performance improvement exceeds gains reported for LSTM-based approaches in building energy forecasting, which typically achieve 15-30% error reduction (Fan et al., 2017)[13], and surpasses the 12% improvement demonstrated by spatial-temporal graph networks in transportation applications (Yu et al., 2018)[15]. This validates the hypothesis that spatial dependencies dominate short-term building dynamics.

The architectural simplicity of the two-layer GCN proves sufficient for short-horizon forecasting while maintaining interpretability. The graph structure makes explicit which sensors influence each prediction through the neighborhood connectivity pattern, enabling operators to understand and validate the system's forecasts. While validated on a 52-sensor building network, the methodology generalizes to larger deployments. The GCN's computational complexity scales linearly with edge count rather than quadratically with node count; a 500-node network with an average degree of five would require only ten times the computation, remaining within edge-computing capabilities.

6.3. Limitations and Constraints

Static Graph Assumption: The fixed topology assumes constant spatial relationships, requiring complete retraining when buildings reconfigure. Potential solutions include meta-learning for quick adaptation or adaptive adjacency matrices that discover latent connections beyond physical proximity.

Short Temporal Window: The 93-second horizon addresses only reactive control, not the 1-24 hour planning required for optimal start/stop. Extending horizons necessitates incorporating weather forecasts, occupancy schedules, and HVAC control signals with hourly/daily cycles, requiring LSTM or Transformer temporal modeling.

Single-Building Evaluation: All results derive from one 54-sensor network. Generalization to other building types, climates, or sensor densities remains unvalidated. Buildings differ in envelope quality, HVAC architecture, and occupancy dynamics. Validating on diverse datasets would test cross-building generalization.

Hardware Constraints: The model requires ~18 KB of memory and ~50,000 floating-point operations per inference, exceeding low-end sensor node capabilities (2-4 KB RAM, 16 MHz processors). Practical deployment requires edge-cloud hybrid architectures (gateway-based inference) or model quantization (8-bit weights, 4× memory reduction, <2% accuracy loss).

Uncertainty Quantification: Point forecasts lack principled uncertainty estimates. The ensemble approach (Figure 20) is computationally expensive (5× training time) and doesn't distinguish epistemic versus aleatoric uncertainty. Safety-critical applications require Bayesian neural networks or conformal prediction methods providing actionable confidence intervals.

6.4. Methodological Reflections

This thesis demonstrates that simple architectures can match complex alternatives when properly matched to problem structure. The two-layer GCN succeeds by encoding spatial locality and thermal inertia architecturally via graph structure and short temporal windows. Complex models like LSTMs must learn these patterns from data, requiring more parameters and larger datasets. By "hard-coding" domain knowledge, the GCN achieves superior data efficiency: 1,376 training samples suffice where LSTMs might require 10,000+. This principle, simplicity through structure, generalizes to many engineering domains exhibiting known physical constraints.

Standard evaluations reporting aggregate metrics obscure spatial heterogeneity. The per-sensor analysis revealed average performance masks a 4.9× range with critical deployment implications. Poor peripheral performance reflects physical reality; boundary conditions are inherently more variable, enabling targeted interventions: avoid critical control sensors at boundaries, sample peripheral sensors more frequently, or combine GCN predictions with physics-based boundary models. Future IoT forecasting studies should routinely include spatial performance breakdowns to guide practical deployment. These methodological reflections bridge directly into the following chapter, which outlines future research directions.

Chapter 7: Conclusion and Future Work

This thesis investigated the application of GNNs for temperature forecasting in IoT sensor networks, addressing a critical gap in building management systems where traditional methods fail to capture spatial dependencies between sensors. Through experimentation on the Intel Berkeley Research Lab dataset, this work demonstrates that explicitly modeling network topology yields substantial performance improvements while maintaining computational efficiency suitable for real-world deployment.

7.1. Answers to Research Questions

RQ1: Performance Superiority: *Can Graph Convolution Networks outperform traditional methods by explicitly modeling spatial dependencies?*
Yes. The GCN achieves 73.16% MSE improvement over Random Forest (0.182 vs. 0.679) and demonstrates 8.9× higher correlation ($r = 0.859$ vs. $r = 0.096$), with statistical significance confirmed via paired t-test ($p < 0.0001$). This superiority stems from the GCN's message-passing mechanism capturing spatial dependencies that traditional methods ignore. The performance gap confirms that explicit topology modeling is essential for networked sensor forecasting.

RQ2: Feature and Configuration Influence: *Which input features and graph configurations most influence forecasting performance?*
Temperature dominates; removing it causes 170% MSE degradation, while eliminating voltage causes only a 39% increase. The 5-nearest-neighbor graph with inverse-distance edge weights balances connectivity and sparsity (116 edges for 52 nodes). Hyperparameter optimization revealed optimal configuration: 32 hidden units, zero dropout, learning rate 0.03, and weight decay 5e-4. The model reaches effective capacity at 60-70% of training data (~1,200-1,400 windows), encouraging for applications with limited historical data.

RQ3: Spatial Performance Heterogeneity: *How does prediction accuracy vary across sensors with differing spatial positions and connectivity?*
Performance varies systematically with network topology. Central sensors with dense connectivity (5-7 neighbors) achieve 4.9× lower error than peripheral sensors with sparse neighborhoods (2-3 neighbors). Sensor 23 (central, MAE = 0.105, $r = 0.996$) vastly outperforms Sensor 49 (peripheral, MAE = 0.518, $r = 0.689$). Edge sensors experience higher variance due to boundary exposure (solar radiation, external airflow), while interior sensors benefit from thermal buffering. This spatial gradient directly informs deployment strategies: prioritize dense instrumentation in critical zones, accept higher uncertainty in peripheral regions.

RQ4: Practical Deployment Principles: *What design principles can guide real-world IoT deployment?*
Five actionable principles emerge: (1) Prioritize spatial coverage over temporal resolution; adding sensors to sparse regions improves accuracy more than increasing sampling rates. (2) Exclude voltage in bandwidth-constrained scenarios, saving 25% transmission overhead with <5% accuracy loss. (3) Apply special handling at boundaries, increase sampling frequency, or combine predictions with physics-based models. (4) Use 70/15/15 train/val/test splits, balancing learning and generalization. (5) Deploy edge-cloud hybrid architectures with gateway-based inference for sub-second latency.

7.2. Key Contributions and Implications

This thesis makes four principal contributions that advance the theoretical understanding and practical application of graph-based forecasting for IoT sensor networks.

First, it provides a **methodological framework** that demonstrates how to transform irregular, multivariate IoT sensor data into graph-structured inputs suitable for neural network forecasting. The complete pipeline, encompassing K-nearest-neighbor graph construction, sliding-window temporal segmentation, and per-sensor normalization, serves as a replicable template for similar spatial-temporal forecasting problems across domains.

Second, the **empirical validation** establishes that spatial dependencies, when explicitly modeled through graph convolutions, yield transformative performance improvements. The 73.16% error reduction over classical baselines, combined with strong correlation ($r = 0.891$) and statistical significance ($p < 0.0001$), confirms that topology-aware learning is not merely incremental but essential for networked sensor forecasting.

Third, the **spatial diagnostic framework** reveals that aggregate metrics alone are insufficient for deployment planning. The 4.9× performance variation between central and peripheral sensors demonstrates that real-world accuracy depends critically on network position, connectivity, and local environmental exposure. This finding shifts the evaluation paradigm from "how well does the model perform?" to "where does the model perform well, and why?"

Fourth, the work provides **actionable deployment principles** grounded in empirical evidence. The identification of temperature as the dominant feature (170% degradation when removed), the stability of 70/15/15 data splits, and the minimal impact of voltage (39% degradation) translate directly into design decisions for practitioners: which sensors to prioritize, which features to transmit under bandwidth constraints, and how to allocate computational resources across the network.

Beyond these technical contributions, the thesis advances a broader methodological principle: **simplicity through structure**. By encoding known physical relationships, spatial proximity, and thermal inertia directly into the model architecture, the GCN achieves superior performance with moderate data requirements, contradicting the assumption that complex problems necessitate complex models. This principle extends beyond IoT forecasting to any domain where system structure can inform model design.

The economic and operational implications discussed in Section 6.2 underscore that these improvements are not purely academic. Energy savings of 10-25%, predictive maintenance capabilities, and scalability to larger deployments position GNN-based forecasting as a practical technology ready for operational building management systems, with clear paths toward district-scale and urban applications.

7.3. Future Research Directions

Dynamic Graph Neural Networks: Current GNN architectures assume fixed topology, failing when sensors relocate or fail permanently. Dynamic GNNs with learned temporal adjacency, where edge weights evolve based on recent correlation patterns, could maintain accuracy without retraining. Graph Attention Networks (GATs) offer one approach, learning attention coefficients that downweight poorly correlated neighbors. The challenge lies in distinguishing meaningful topological evolution from transient noise correlation, requiring physics-informed regularization that penalizes edges between physically distant sensors.

Explainable AI for Trust and Adoption: Facility managers resist "black-box" forecasts when HVAC decisions affect occupant comfort and energy budgets. Attention weight visualization showing which neighbors influenced each prediction, counterfactual analysis revealing what-if scenarios, and sensitivity maps indicating forecast confidence across building zones could bridge this trust gap. GNNExplainer methods identify critical subgraphs responsible for predictions, enabling diagnostics that operators can validate against domain knowledge.

Edge Deployment and Model Compression: Deploying GNNs on sensor nodes (rather than gateways) enables fully decentralized forecasting with improved robustness to gateway failures. This requires aggressive compression: quantization (8-bit weights, 4× memory reduction, <2% accuracy loss), pruning (remove 50-70% of connections with <3% degradation), and knowledge distillation (small "student" models mimicking large "teacher" outputs).

7.4. Societal Impact and Sustainability

Buildings account for 40% of global energy consumption and 33% of greenhouse gas emissions. If demonstrated 10-25% savings scaled to just 10% of commercial buildings worldwide (20 billion m^2 floor space), annual savings would reach 40-100 TWh, preventing 20-50 million tonnes of CO_2 emissions, equivalent to removing 4-10 million cars from roads. However, environmental calculus must account for sensor manufacturing (rare earth metals, energy-intensive fabrication) and e-waste. Lifecycle assessments suggest networks achieve carbon neutrality within 1-2 years through energy savings, but responsible deployment requires end-of-life recycling programs and design-for-disassembly principles.

High-resolution temperature sensing enables occupancy inference, potentially revealing sensitive information about meeting patterns and workspace usage. Robust data governance must enforce purpose limitation (sensor data used solely for HVAC control), minimize retention periods (7-30 days sufficient for training), and provide occupant transparency regarding collection and use. Differential privacy techniques, adding calibrated noise to readings, could formally bound re-identification risk while maintaining forecast accuracy.

Advanced building management concentrates on premium commercial real estate, leaving public housing, schools, and small businesses with legacy controls. This efficiency gap compounds economic inequality; low-income households spend 7-10% of their income on energy versus 2-3% for affluent households. Democratizing GNN-based forecasting requires open-source implementations, pre-trained models transferring across building types with minimal tuning, and low-cost sensor packages. Partnerships between municipalities, universities, and community organizations could deploy pilots in underserved buildings, measuring not only energy savings but also occupant satisfaction and health outcomes.

7.5. Summary

This thesis demonstrates that Graph Neural Networks provide a principled, effective framework for IoT sensor forecasting by explicitly encoding network topology as inductive bias. The 73% performance improvement over classical baselines is transformative, elevating forecasting from barely-better-than-random correlation to strong predictive capability suitable for closed-loop control. The simplicity of the two-layer architecture, requiring only 18 KB of memory, proves that matching model structure to problem structure enables efficiency without sacrificing accuracy.

Beyond technical contributions, this work provides a template for spatial-temporal forecasting in any domain where sensors exhibit network structure: traffic prediction on road networks, air quality monitoring across urban grids, precision agriculture with soil moisture arrays, and industrial process control in manufacturing plants. The methodology generalizes: construct k-NN graphs from spatial coordinates or learned correlation structure, apply graph convolution to capture neighborhood influence, and combine with temporal modeling for long-range forecasting.

The path forward requires interdisciplinary collaboration, machine learning researchers advancing dynamic GNN architectures, building scientists validating forecasts against first-principles models, policymakers establishing data governance frameworks, and facility managers providing operational feedback grounding research in real constraints. As IoT deployments accelerate toward 40 billion connected devices by 2034, the question is not whether graph-based learning will become standard practice, but how quickly we can transition from research prototypes to production systems delivering measurable energy savings, cost reductions, and environmental benefits at a global scale.

References

[1] Statista (2025). Internet of Things (IoT) connections worldwide from 2022 to 2023, with forecasts from 2024 to 2034 | https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/

[2] Intel Berkeley Research Lab Sensor Data | https://www.kaggle.com/datasets/divyansh22/intel-berkeley-research-lab-sensor-data/data

[3] El Husseini et al. (2025). Machine Learning in Smart Buildings: A Review of Methods | https://www.mdpi.com/3395530

[4] Kipf & Welling (2017). Semi-Supervised Classification with Graph Convolution Networks | https://arxiv.org/abs/1609.02907

[5] Gubbi et al. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions | https://www.sciencedirect.com/science/article/abs/pii/S0167739X13000241

[6] Elsts et al. (2018). Temperature-resilient time synchronization for IoT | https://inria.hal.science/hal-01706802/file/elsts18temprature-resilient-tsch.pdf

[7] Pérez-Lombard et al. (2008). A review on buildings' energy consumption information | https://www.sciencedirect.com/science/article/abs/pii/S0378778807001016

[8] Box et al. (2016). Time Series Analysis: Forecasting and Control | http://repo.darmajaya.ac.id/4781/1/Time%20Series%20Analysis_%20Forecasting%20and%20Control%20%28%20PDFDrive%20%29.pdf

[9] Hyndman & Athanasopoulos (2021). Forecasting: Principles and Practice | https://otexts.com/fpp3/

[10] Tibshirani (1996). Regression Shrinkage and Selection via the Lasso | Source

[11] Breiman (2001). Random Forests | https://link.springer.com/article/10.1023/A:1010933404324#preview

[12] Pedregosa et al. (2011). Scikit-learn: Machine Learning in Python | https://jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf

[13] Fan et al. (2017). Deep recurrent neural network-based strategies for short-term building energy predictions | https://www.sciencedirect.com/science/article/abs/pii/S0306261917302921?via%3Dihub

[14] Wu et al. (2019). A Comprehensive Survey on Graph Neural Networks | https://arxiv.org/pdf/1901.00596

[15] Yu et al. (2018). Spatio-Temporal Graph Convolution Networks: A Deep Learning Framework for Traffic Forecasting | https://arxiv.org/pdf/1709.04875

[16] Li et al. (2018). Diffusion Convolution Recurrent Neural Network: Data-Driven Traffic Forecasting | https://arxiv.org/pdf/1707.01926

[17] Wu et al. (2019). Graph WaveNet for Deep Spatial-Temporal Graph Modeling | https://arxiv.org/pdf/1906.00121

Appendix A: Colab and GitHub Repository

- **Google Colab Notebook**: GNN_IoT_Temperature_Forecasting_Study_vFinal.ipynb
- **GitHub Repository**: GNN-IoT-Temperature-Forcasting-Study

Appendix B: Acronyms List

Acronym	Full Form
AI	Artificial Intelligence
CNN	Convolutional Neural Network
DCRNN	Diffusion Convolution Recurrent Neural Network
GAT	Graph Attention Network
GCN	Graph Convolution Network
GNN	Graph Neural Network
HVAC	Heating, Ventilation, and Air Conditioning
IoT	Internet of Things
IQR	Interquartile Range
KPI	Key Performance Indicator
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MLP	Multi-Layer Perceptron
MPC	Model Predictive Control
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
STGCN	Spatio-Temporal Graph Convolution Network

Appendix C: Hardware and Software Environment Summary

Category	Specification
Hardware	Intel Core i7 CPU @ 2.6 GHz, 16 GB RAM
GPU	NVIDIA T4 (Colab)
Operating System	macOS 14
Python Version	3.11
Python Libraries	PyTorch 2.3.0, PyTorch Geometric 2.5.3, NumPy 1.26, Pandas 2.2, Matplotlib 3.8, Seaborn 0.13
Environment	Google Colab (GPU Runtime)
Version Control	Git (GitHub repository or version tracking)