Temporal Graph Modeling for Sparse Maritime Networks

Abstract

This paper presents a self-contained approach to temporal graph modeling tailored for sparse maritime port networks, with a specific application to the Great Lakes–St. Lawrence (GLSL) corridor. Despite the critical role of these inland waterway systems, decision-making is often hampered by data sparsity and intermittent traffic patterns that standard predictive models fail to capture. Leveraging Automatic Identification System (AIS) data spanning 12 years, we propose a methodology that addresses these challenges through spatial node aggregation and a novel temporal k-core decomposition algorithm. We conduct a comparative analysis of three deep learning architectures: a baseline Long Short-Term Memory (LSTM) network, a sequential Graph Attention Network integrated with a Gated Recurrent Unit (GAT-GRU), and a parallel Graph WaveNet model. Our empirical results indicate that the GAT-GRU architecture offers superior predictive performance in this sparse setting, outperforming both the LSTM baseline and Graph WaveNet. Beyond forecasting, we demonstrate how this framework enables counterfactual analysis to simulate network responses to exogenous shocks, providing a robust foundation for strategic planning and resilience assessment in maritime supply chains.

Keywords Maritime Networks · Temporal Graph Neural Networks · Great Lakes Saint-Lawrence Corridor · AIS Data

Declaration of Competing Interest. *The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.*

Data. *The data used in this paper were obtained and utilized under the conditions of a data sharing agreement prohibiting its public release or redistribution.*

1 Introduction

This paper aims to provide both a methodological and pedagogical contribution by proposing a self-contained approach to temporal graph modeling, specifically tailored for sparse maritime port networks. Decision-makers in the maritime sector face significant challenges when planning infrastructure investments, particularly in environments where data is thin or activity is intermittent. In these relatively inactive port networks, standard predictive models often fail to capture the underlying dynamics necessary for robust strategic planning.

Despite substantial advances in vessel traffic flow prediction, significant methodological gaps persist. A central challenge is data sparsity—a condition where a large share of potential interactions is either weakly observed or missing [1]. In maritime networks, sparsity emerges from the highly uneven distribution of traffic intensity and vessel movements across ports. This geospatial heterogeneity complicates the construction of reasonable port network representations and the modeling of spatiotemporal dependencies. Yet, despite its practical importance, sparsity has received limited attention in existing literature, which predominantly focuses on highly active systems like the Shanghai maritime network [2]. Consequently, less active port networks remain inherently more challenging to analyze.

A second critical gap concerns the scarcity of counterfactual analysis frameworks in current maritime studies. This deficiency is often tied to the lack of relevant feature spaces in the graph structures being learned; many existing models rely solely on in-and-out node flows (e.g., [2]), which limits their explanatory power. While prior research has prioritized predictive accuracy, few studies examine how traffic dynamics would evolve under hypothetical scenarios. This limitation reduces the utility of such models for policy evaluation and strategic planning, particularly in navigation systems that are structurally exposed to diverse disturbances.

1.1 The GLSL Maritime Corridor

We define the Great Lakes–St. Lawrence (GLSL) maritime corridor as the maritime network extending from Duluth, Minnesota (western terminus) to Sept-Îles, Québec (eastern terminus). The GLSL encompasses the five Great Lakes (Superior, Michigan, Huron, Erie, and Ontario) and the St. Lawrence Seaway, forming a critical inland waterway system that connects North American industrial centers to global markets [3]. This is especially true in the current context, as both U.S. and Canadian firms engaged in cross-border trade are seeking to diversify. Indeed, newly imposed tariffs are likely to increase demand for maritime trade with international partners. This raises questions about the GLSL corridor's readiness to accommodate future demand, particularly given increasingly volatile water levels in the St. Lawrence River. Climate-change impacts on St. Lawrence River hydrology may, in turn, disrupt navigation and supply chains throughout the corridor.

In this paper, we employ Automatic Identification System (AIS) data and design a methodology that jointly captures structural and temporal dependencies to learn a holistic representation of the system while explicitly addressing the issue of data sparsity. More specifically, we train Graph WaveNet [4] as well as Temporal Graph Attention Networks (TGAT) [5, 6], both powerful architectures, alongside a baseline Long Short Term Memory (LSTM) [7] model to predict future network states. One of the main results is that we concluded that GAT with Gated Recurrent Unit (GRU) was superior to graph WaveNet with a default set of hyperparameter and no fine tuning on our AIS dataset of the GLSL data.

1.2 Related Work

The predominance of Graph Neural Network (GNN) based methods in the general topic of traffic flow modeling is well-established in the existing literature [8, 9, 10]. This popularity can be explained by the unique capability of these models to encompass jointly the spatial interdependencies and temporal dynamics, which are intrinsic to the transportation system.

However, in the specific domain of port network analysis and maritime traffic, a variety of methodological approaches have been proposed. A notable example is the shipping route optimization method based on network feature of ports [11], which aims to enhance port resilience by improving the shipping route network, using a link-prediction-based approach. This research offers a valuable approach for the identification of alternative network connections, particularly in the context of disruptions caused by natural disasters. Nevertheless, such approach is based on a static data representation of the shipping network, without explicitly modeling the temporal dynamics of traffic flows. As a consequence, predicted outcomes are treated as equally feasible regardless of timing or traffic intensity, limiting their applicability for an empirical and more realistic forecasting.

Most of the traditional maritime transport literature has mainly relied on static representations of port networks [12]. Advanced modeling techniques such as spatial-temporal graph modeling and neural networks have significantly shifted the way maritime traffic flows can be predicted. In particular, state-of-the-art methods based on Graph Wave Net architectures have demonstrated strong abilities to capture long-term temporal sequences and the latent spatial relationships embedded in the traffic flow data [4].

Within the maritime domain, several studies have applied spatiotemporal GNN-based models to vessel traffic flow prediction [13, 14, 2, 15]. By leveraging AIS records, such models are able to build multi-graph convolutional networks for traffic flow prediction, enabling the extraction of both spatial and temporal traffic flow dimensions [13].

Spatiotemporal GNN-based approaches have also proven particularly valuable in contexts characterized by exogenous shocks in the maritime supply chain, such as the COVID-19 pandemic [15]. By employing a spatiotemporal dynamic graph neural network (STDGNN), regional vessel flows can be effectively predicted during large-scale disruptions such as pandemics. This evidence demonstrates the importance for decision-makers to adopt prediction-driven policy frameworks, which contribute to strengthen the operational resilience of global maritime networks against future shocks.

1.3 Paper Layout

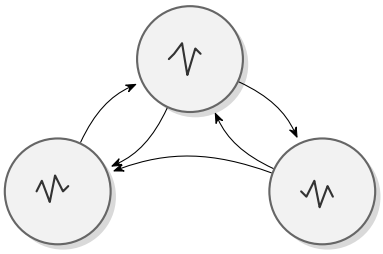
The remainder of this paper is organized as follows. Section 2 provides the theoretical background on spatiotemporal graph modeling and details the architectures of the GAT-GRU and Graph WaveNet models. Section 3 presents our methodology for modeling the GLSL maritime network, including the problem definition, our approach to mitigating data sparsity via node aggregation and temporal k-core decomposition, and the comparative performance analysis of the trained models. Section 4 discusses the implications of our findings, introduces a framework for counterfactual analysis, and outlines future research directions.

2 Background

Spatiotemporal graph modeling. In spatiotemporal graph modeling, the goal is to define a model that can represent a topological graph structure for which each node has dynamic input

features, as depicted in Figure 1. This paradigm extends beyond static graph analysis by requiring the simultaneous processing of structural connectivity and time-series evolution. Spatiotemporal Graph Neural Networks (STGNNs) have emerged as the state-of-the-art method for modeling such dynamic systems on non-Euclidean structures, including traffic networks and supply chains. While foundational Graph Neural Networks (GNNs) generalized convolutions to graph-structured data by aggregating neighbor information [16, 17], they lack mechanisms to process time-series data inherently. To bridge this gap, early advancements such as the Spatiotemporal Graphical Convolutional Network (GCN) [18] integrated graph convolutions with recurrent neural networks (RNNs) or 1D convolutions. In this work, we focus on two distinct evolutionary branches of this technology: the recurrent-based GAT-GRU and the convolution-based Graph WaveNet. Figure 2 presents a detailed schematic comparison of their internal mechanisms.

Figure 1: Temporal graph modeling.



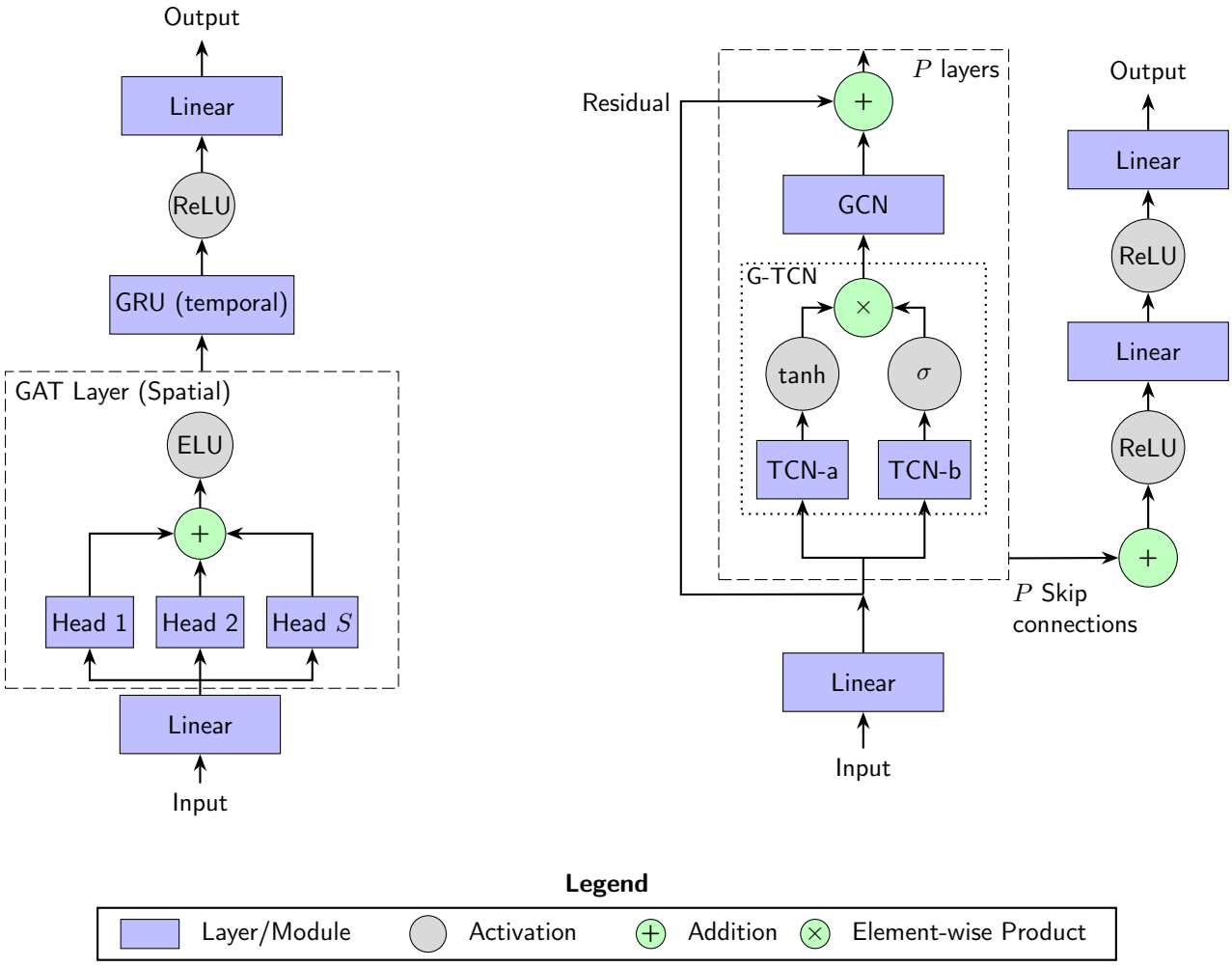
GAT-GRU. This architecture represents a "sequential" approach to spatiotemporal modeling, integrating Graph Attention Networks (GATs) [5] within a Gated Recurrent Unit (GRU) [6] framework. As illustrated in Figure 2 (left), the input features are first projected through a linear layer before entering the spatial block. The GAT layer employs a multi-head attention mechanism (Head 1 to Head S) to dynamically weigh neighbor importance, aggregating these representations via summation ($+$) followed by an ELU activation. This spatially enriched signal is then passed to the GRU layer, which manages temporal dependencies. Finally, the output stack processes the GRU hidden states through a $ReLU$ activation and a final Linear layer to generate the prediction.

Graph WaveNet. Graph WaveNet [4] represents a "parallel" approach designed to overcome the computational bottlenecks of recurrent structures. As shown in Figure 2 (right), the architecture is built upon a stack of p layers containing a Gated Temporal Convolutional Network (G-TCN) and a Gated Convolutional Network (GCN) layer. The G-TCN integrates a 1-dimensional causal convolution that handles temporal flow, while the GCN handles the spatial relationships. The G-TCN splits the input into two branches (TCN-a and TCN-b) processed by tanh and sigmoid (σ) activations, respectively, which are then combined via an element-wise product ($\times$) to control information flow. The output is further processed by a GCN layer and a residual connection ($+$) to preserve signal depth. Crucially, the model utilizes skip connections from each layer, which are summed ($+$) and passed through a deep output stack consisting of alternating ReLU activations and linear layers to produce the final output.

3 Modeling the GLSL Maritime Network

Our data is composed of AIS-derived origin–destination records spanning from January 2013 to December 2024, and includes only trips where at least one endpoint (origin or destination) is a

Figure 2: Algorithmic architectures of GAT-GRU (left) and Graph WaveNet (right).



GLSL port. The data captures vessel movements to, from, and throughout the GLSL maritime corridor. This includes 1,265 non-GLSL ports interacting with 87 GLSL ports. The final data table totals 409,882 trips over the 12-year period, involving 1,352 distinct ports and 7,446 commercial vessels.

3.1 Problem Definition

We define a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from this data by adapting ports as nodes and trips as edges. The set of nodes and edges are respectively given by $\mathcal{V} = \{v_1, \ldots, v_n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Each datapoint contains characteristics about ports, vessels, and trips. Therefore, we define $X_f \in \mathcal{R}^{n \times m}$ and $X_e \in \mathcal{R}^{n^2 \times l}$ as the matrices for node and edge features, as well as $A \in \mathcal{B}^{n \times n}$, the adjacency matrix. The dimensions n, m, and l, respectively represent the number of nodes, node features, and edge features.

It is important to select a set of features that can be useful in a counterfactual framework setting, so that decision-makers can use trained models to simulate the impact of a simulated shock on relevant attributes. For instance, by editing the values of the features shown in Table 1, one can

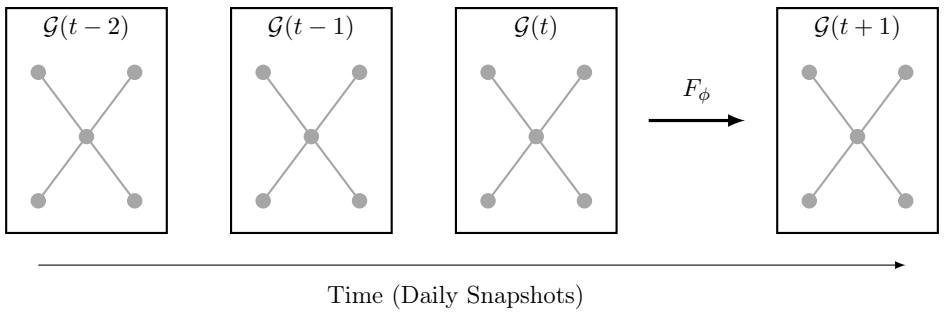
Table 1: Description of the attributes.

Level	Attributes	Description
Node	Inbound flows	Number/volume of incoming trips
	Outbound flows	Number/volume of outgoing trips
	Dwell time	Average time vessels spend at the port
	Deadweight	Sum deadweight tonnage of vessels
	Speed over ground (SOG)	Average speed of vessels entering/leaving the port
Edge	Trip duration (avg)	Average trip duration
	Trip duration (std)	Standard deviation of trip durations
	Trip count	Number of trips on an edge

simulate port congestion or strikes (dwell time and SOG), load rerouting (deadweight), or closing routes (rebalancing deadweight with in and out flows).

So far, we have defined a master graph $\mathcal{G}$ over the whole aggregated 12-year period. However, in order to predict the next state of the network from recent activity, we need to derive subgraphs $\mathcal{G}(t)$ with $t \in \{1, \ldots, T\}$ representing snapshots of the network at time t. Therefore, each datapoint $\mathcal{G}(t)$ is a collection $\{X_f(t), X_e(t), A(t)\}$.

Figure 3: Problem definition.



As depicted in Figure 3, we aim to optimize the parameters ϕ of a function F that maps the next state of the graph $\mathcal{G}(t+1)$ from a chronological input sequence of graphs $\{\mathcal{G}(t-j), \mathcal{G}(t-j+1), \ldots, \mathcal{G}(t)\}$ of length j.

At this point, an important consideration arises—time granularity. Indeed, there is a trade-off between time granularity and data sparsity. Some GLSL ports are not very active, and some other non-GLSL nodes rarely interact with the GLSL subnetwork, meaning that the data is inherently sparse. Sparsity naturally increases as T increases.

Unfortunately, time aggregation cannot be used to mitigate sparsity as, for instance, by using weekly snapshots we negatively impact the number of training examples. The proper balance in terms of time granularity depends on the amount of data available, as deep learning models scale with it. In our case, weekly graph snapshots would provide an underwhelmingly small training set, so we instead opt for daily snapshots.

3.2 Mitigating Sparsity

Using daily GLSL port network snapshots, we obtain a sequence of 4,379 graphs spanning 12 years of activity. A significant challenge in this setting is that the probability of a node showing non-zero feature values on any given day is highly variable. Many ports are structurally present in the master graph but operationally inactive for long periods, acting as "ghost nodes" that introduce noise (zeros) into the training data.

Our sparsity mitigation strategy is twofold: we first apply (a) spatial aggregation to non-GLSL nodes, and then (b) we formulate a temporal filtering algorithm that we apply on the aggregated graphs. Both are needed because (a) alone would fail to eliminate weak signals emanating from mostly inactive ports within the GLSL network, and (b) alone would be too restrictive as most of the exchanges would be pruned as only very few GLSL and international ports interact consistently. This arrangement allows us to eliminate sparsity while preserving most of the information contained in our graph sequence.

Node aggregation. We create a mapping from port locations to their geographic regions. We divide the world into Oceania, Europe, Asia, Africa, North and South America. The aggregation is applied to non-GLSL nodes since we devise a methodology that focuses on a specific port network. This kind of aggregation reduces the number of nodes as well as the total degree of non-aggregated nodes (the GLSL ports) but significantly increases feature density. This tradeoff works in our favor because the exact location of international partners does not constitute crucial information.

Temporal k-core decomposition. We mitigate this data sparsity by formulating a k-core decomposition-based algorithm [19] that takes temporal availability into account. Our method minimizes sparsity while maximizing coverage under the constraint that the subgraphs must be stable in their structure through time. In other words, we aim to maximize the density of non-zero features while ensuring the resulting subgraph remains structurally cohesive, a prerequisite for the spatial convolution layers in Graph WaveNet.

Algorithm 1 shows how our temporal k-core decomposition algorithm filters nodes based on their connectivity patterns over time. Consider a temporal graph $\mathcal{G} = \{G_1, G_2, \ldots, G_T\}$ where $G_t = (V_t, E_t)$ represents the graph at time t. The k-core of a graph $G = (V, E)$ is the maximal subgraph $G_k = (V_k, E_k)$ where each node $v \in V_k$ has degree $\deg(v) \geq k$.

The k-core is the maximal subgraph where each node has at least k connections. For each time step t, we compute the k-core decomposition to identify densely connected nodes. We track the frequency $f(v)$ at which each node v appears in the k-core across all time steps. Nodes with frequency $f(v) \geq h$ are retained in the final filtered graph. This identifies ports that maintain strong connectivity patterns over time. The output is a filtered temporal graph dataset compatible with PyTorch Geometric, enabling prediction tasks on maritime traffic patterns.

3.3 Training and Performance

After applying Algorithm 1, we obtain 4,383 temporal graph instances. Input-target sequence pairs were constructed using a sliding window approach with a sequence length of 12 timesteps. For each sequence, the model receives 12 consecutive graph snapshots as input and predicts the graph-level representation of the subsequent timestep. The training process was conducted with the set of hyperparameters described in Table 3.

Algorithm 1 Temporal K-Core Decomposition

Require: Temporal graph $\mathcal{G} = \{G_1, \ldots, G_T\}$, k-core parameter k, frequency threshold $h \in [0, 1]$
Ensure: Filtered node set V^*

Initialize $\mathcal{A} \leftarrow \{\}$ ▷ Collection of active node sets
for $t = 1$ to T **do**
  $G_t \leftarrow (V_t, E_t)$ ▷ Get graph at time t
  if $k \leq 0$ or k is None **then**
    $A_t \leftarrow V_t$ ▷ All nodes are active
  else
    Remove self-loops from G_t
    $G_k^t \leftarrow \text{K-Core}(G_t, k)$ ▷ Compute k-core subgraph
    $A_t \leftarrow \text{nodes}(G_k^t)$ ▷ Extract k-core nodes
  end if
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{A_t\}$
end for
▷ Calculate temporal frequency for each node
for each node $v \in \bigcup_{t=1}^{T} V_t$ **do**
  $f(v) \leftarrow \frac{1}{T} \sum_{t=1}^{T} \mathbb{1}_{v \in A_t}$ ▷ Frequency of v in active sets
end for
$V^* \leftarrow \{v : f(v) \geq h\}$ ▷ Filter nodes by threshold
return V^*

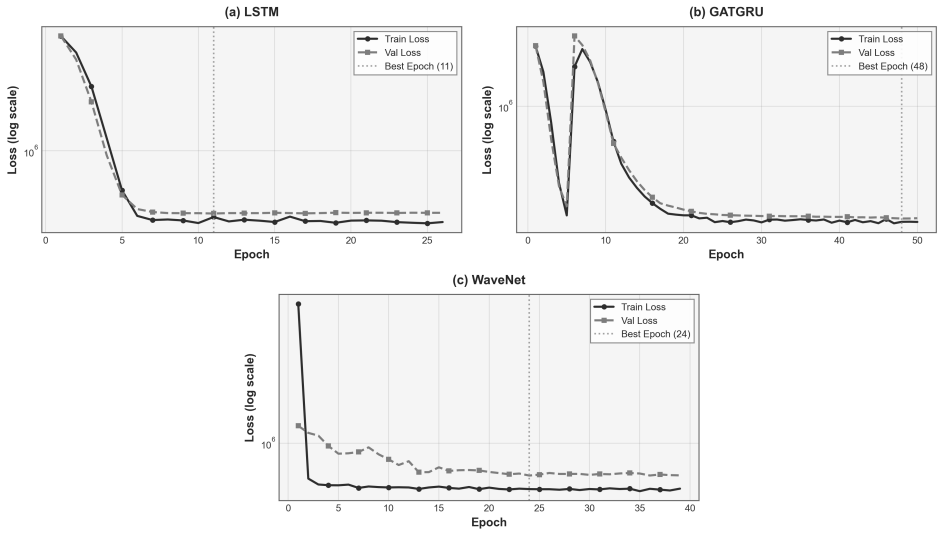
Figure 4 illustrates the training dynamics and convergence properties of the three comparative models. All architectures demonstrate stable learning behaviors with monotonic decay in loss after initial stabilization. The LSTM baseline (Figure 4(a)) exhibits rapid initial convergence, reaching its best validation loss early at epoch 11. In contrast, the graph-based architectures show distinct behaviors. The GAT-GRU (Figure 4(b)) exhibits significant volatility in the initial epochs but stabilizes effectively to achieve a robust descent, reaching its best validation performance at epoch 48. Graph WaveNet (Figure 4(c)) converges relatively quickly with its best epoch recorded at 24, though the gap between training and validation loss suggests a higher susceptibility to overfitting compared to the other models. Early stopping (indicated by vertical dotted lines) was deployed to retain the most generalizable weights.

Table 2: Comparative performance of temporal graph models on test set. Performance metrics are given in their standardized forms.

Model	MAE	RMSE	Parameters	Training Epochs	Training Time (s)
LSTM	375.41	603.15	88,000	26	1,997
GAT-GRU	**292.20**	**435.06**	80,384	50	6,956
WaveNet	411.83	886.96	221,376	39	5,946

As shown in Table 2, the GAT-GRU model demonstrates superior performance across both evaluation metrics, achieving the lowest MAE and RMSE on the test set. This represents a significant improvement over the baseline LSTM model and the Graph WaveNet. The enhanced performance of GAT-GRU suggests that the attention mechanisms are successfully capturing the relevant spatial dependencies between nodes, which, when combined with the GRU's temporal processing, yields the most precise predictions. Surprisingly, Graph WaveNet performed the poorest among

Figure 4: Mean Square Error (MSE) training and validation losses over training epochs given in log scale for LSTM, GAT-GRU and graph WaveNet.



the three models, with the highest error rates, indicating that its dilated causal convolutions may not have been as effective for this specific dataset configuration compared to the attention-based approach.

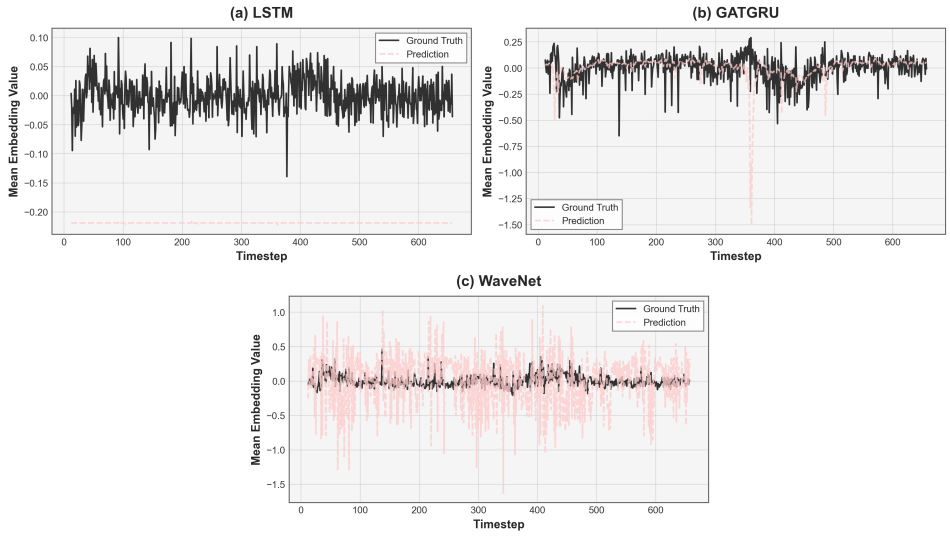
Regarding computational efficiency, the models present a trade-off between parameter size, training time, and accuracy (see Table 2). The GAT-GRU model possesses the fewest parameters yet requires the longest training time, likely due to the computational complexity of calculating attention scores across the graph structure. Conversely, Graph WaveNet has the highest parameter count and a training time inferior to that of GAT-GRU. The LSTM baseline remains the most computationally efficient option, but lacks the structural understanding of the GAT-based model. However, it is important to note that these results highly depend on hyperparameters, as these models are very sensitive to them. Consequently, results could differ with a different set of hyperparameters or even on a different data distribution, as these models are also very dependent on data. The hyperparameters we used are shown in Table 3 in the Appendix.

The qualitative differences in model performance are further visualized in Figure 5. The LSTM model (Figure 5(a)) fails to capture the spatial dynamics, producing a flat-line prediction that approximates the mean of the target variable. Graph WaveNet (Figure 2(c)), while attempting to model the fluctuations, exhibits excessive variance and noise, leading to the high error rates observed in Table 2. In contrast, GAT-GRU (Figure 2(b)) successfully tracks the ground truth signal with reasonable fidelity, confirming its status as the optimal choice for this task.

4 Discussion

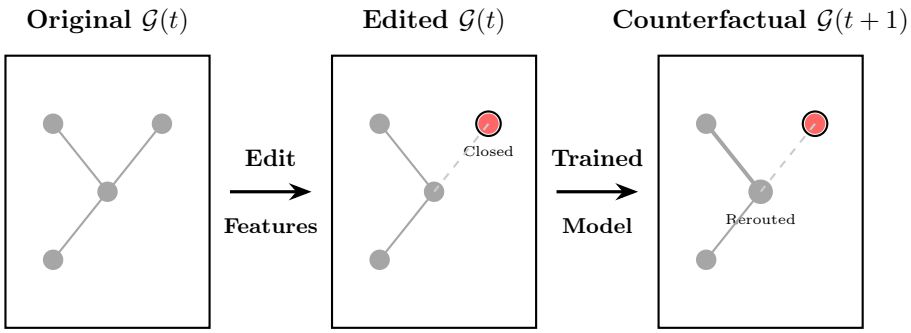
Temporal graph models offers a foundation to further craft useful tools for decision-makers. For instance, it is possible to generate counterfactual predictions based on edited input graph features as shown in Figure 6. Editing an input sequence, or in other words, simulating counterfactual scenarios via intervention in the input data, is done by modifying the relevant values in either the node or edge feature matrix, or both. The now edited graph sequence is fed to the trained model which

Figure 5: Overall inference accuracy, ground truth against predictions for LSTM, GATGRU and Graph WaveNet.



outputs the associated counterfactual prediction for the next period. In the context of maritime traffic, one might want to create shocks on the inbound flows, or dwell time in order to simulate the effect of strikes or speed regulations.

Figure 6: Visualization of the counterfactual analysis framework



Depending on the input sequence length j, it is possible to simulate shocks that propagate over an extensive period of time. However, since j is fixed and defined before training, it is better to know in advance how long the input sequence needs to be. Furthermore, the maximum practical value for j is relative to the total amount of available data points, and excessively long input sequences might hinder training.

In order to predict for an extended sequence, it is possible to do so by recursively adding the new prediction to the input sequence while respecting the chronological order. It is also possible to train the main model on a larger output sequence, but we advise against it if the number of training examples is not large enough. Indeed, it is significantly more difficult to maintain accuracy as the length of the predicted output sequence increases because of *autoregressive drift* (i.e., error accumulation). Even though Graph WaveNet includes mitigation mechanisms, namely

non-autoregressive output and the stacked TCN acting as a feature extractor, the ability to predict long sequences still heavily depends on the training data.

Structural edits (e.g., closing a port as pictured in Figure 6) are possible but more difficult because such intervention needs to be curated also at the edge level, and thus replacing every node feature value of a port by 0 does not suffice. Also, structural edits must take into account the structure of the model that will be used to predict counterfactual outputs. For instance, if a Graph WaveNet architecture is used, a self-adaptive adjacency matrix is learned from the data during training but the adjacency matrix remains unchanged during test time. Graph WaveNet does not generate a dynamic graph that changes at every time step based on the input features unlike the attention mechanism in GAT, which is dynamic and data-dependent.

The lack of cross-validation and model fine-tuning is a limitation of this study. However, we show nonetheless that a priori concepts—such as Graph WaveNet being superior in theory—do not hold all the time, and that GAT-GRU seems the better choice here.

Another crucial consideration is to distinguish between *correlational* and *causal* counterfactuals. The counterfactual predictions that one can generate using the models trained in this study would only reflect learned statistical associations in the training data, not true causal mechanisms. This distinction has important implications for practical applications: while our approach can predict how traffic patterns might change under hypothetical scenarios, it cannot definitively establish whether observed changes would be caused by the interventions or merely correlated with them.

For policy applications—such as evaluating the impact of new port infrastructure or route restrictions—causal inference would be more appropriate. Establishing causality would require either controlled experiments (impractical in maritime contexts) or the application of causal inference frameworks such as do-calculus and structural causal models. Future work should explore integrating these approaches to move beyond predictive counterfactuals toward true causal reasoning.

4.1 Future Directions

We identify three promising and complementary directions for future research.

First, integrating external data sources is essential for enriching the model context. Supplementing graph data with port capacity databases, trade agreements, cargo manifests, and environmental records would provide crucial information currently missing from purely topological approaches. This integration would support more accurate predictions and grounded counterfactual scenarios.

Second, developing a rigorous causal inference framework using do-calculus and structural causal models would transform correlational counterfactuals into causal interventions. This direction involves identifying confounders, establishing causal graphs, and applying identification strategies to estimate true causal effects from observational data, thereby moving beyond statistical association.

Third, framing the maritime network as an environment for reinforcement learning (RL) could enable the optimization of sequential decisions, such as route planning and capacity allocation, under uncertainty. This approach naturally handles the dynamic, multi-step nature of maritime logistics while learning policies that maximize long-term objectives. These directions are mutually reinforcing: external data integration improves the environment model for RL agents, while causal inference ensures that learned policies reflect true causal mechanisms rather than spurious correlations.

References

[1] M. Nasiri, B. Minaei, and Z. Sharifi, "Adjusting data sparsity problem using linear algebra and machine learning algorithm," *Applied Soft Computing* **61** (Dec., 2017) 1153–1159. https://www.sciencedirect.com/science/article/pii/S1568494617303071.

[2] Y. Li, Z. Li, Q. Mei, P. Wang, W. Hu, Z. Wang, W. Xie, Y. Yang, and Y. Chen, "Research on Multi-Port Ship Traffic Prediction Method Based on Spatiotemporal Graph Neural Networks," *Journal of Marine Science and Engineering* **11** no. 7, (July, 2023) 1379. https://www.mdpi.com/2077-1312/11/7/1379.

[3] "Economic Impacts." https://greatlakes-seaway.com/en/the-seaway/economic-impacts/.

[4] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for Deep Spatial-Temporal Graph Modeling," *arXiv e-prints* (May, 2019) arXiv:1906.00121. _eprint: 1906.00121.

[5] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*. 2018.

[6] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734. 2014.

[7] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.* **9** no. 8, (Nov., 1997) 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

[8] S. Wang, Y. Lv, P. Yuan, X. Piao, and Y. Zhang, "Metro Traffic Flow Prediction via Knowledge Graph and Spatiotemporal Graph Neural Network," *Journal of Advanced Transportation* **2022** (2022) . https://login.proxy2.hec.ca/login?url=https://www.proquest.com/scholarly-journals/metro-traffic-flow-prediction-via-knowledge-graph/docview/2720246630/se-2?accountid=11357.

[9] G. Huo, Y. Zhang, Y. Lv, H. Ren, and B. Yin, "Urban Traffic Flow Forecasting Based on Graph Structure Learning," *Journal of Advanced Transportation* **2024** (2024) . https://login.proxy2.hec.ca/login?url=https://www.proquest.com/scholarly-journals/urban-traffic-flow-forecasting-based-on-graph/docview/3151793912/se-2?accountid=11357.

[10] L. Huang, J. Qin, and T. Wu, "Multisource Data Fusion With Graph Convolutional Neural Networks for Node-Level Traffic Flow Prediction," *Journal of Advanced Transportation* **2024** no. 1, (Jan., 2024) 7109780. https://onlinelibrary.wiley.com/doi/10.1155/atr/7109780.

[11] X. Yuan and X. He, "Enhancement Strategy for Port Resilience: Shipping Route Optimization Methods Based on Network Characteristics of Ports," *Journal of Marine Science and Engineering* **13** no. 2, (Feb., 2025) 325. https://www.mdpi.com/2077-1312/13/2/325.

[12] R. Yan, S. Wang, L. Zhen, and G. Laporte, "Emerging approaches applied to maritime transport research: Past and future," *Communications in Transportation Research* **1** (Dec., 2021) 100011. https://www.sciencedirect.com/science/article/pii/S2772424721000111.

[13] M. Liang, R. W. Liu, Y. Zhan, H. Li, F. Zhu, and F.-Y. Wang, "Fine-Grained Vessel Traffic Flow Prediction With a Spatio-Temporal Multigraph Convolutional Network," *IEEE Transactions on Intelligent Transportation Systems* **23** no. 12, (Dec., 2022) 23694–23707. https://ieeexplore.ieee.org/document/9868210/.

[14] F. Zhang, Y. Wen, O. Valdez Banda, M. Chen, and L. Du, "Maritime traffic network extraction and vessel flow prediction in complex inland port clusters," *Ocean Engineering* **341** (Dec., 2025) 122791. https://linkinghub.elsevier.com/retrieve/pii/S0029801825024746.

[15] C. Zhao, X. Li, M. Zuo, L. Mo, and C. Yang, "Spatiotemporal dynamic network for regional maritime vessel flow prediction amid COVID-19," *Transport Policy* **129** (Dec., 2022) 78–89. https://linkinghub.elsevier.com/retrieve/pii/S0967070X22002797.

[16] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks* **20** no. 1, (2009) 61–80.

[17] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems* **32** no. 1, (2021) 4–24.

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

[19] J. I. Alvarez-Hamelin, L. Dall'Asta, A. Barrat, and A. Vespignani, "k-core decomposition: a tool for the visualization of large scale networks," Oct., 2005. http://arxiv.org/abs/cs/0504107. arXiv:cs/0504107.

Appendix

Table 3: Hyperparameters for the three models

Hyperparameter	**LSTM**	**GAT-GRU**	**Graph WaveNet**
Training Configuration			
Train/Val/Test Split	0.7/0.15/0.15	0.7/0.15/0.15	0.7/0.15/0.15
Batch Size	32	32	32
Input Sequence Length	12	12	12
Number of Epochs	50	50	50
Early Stopping Patience	15	15	15
Model Architecture			
Hidden Dimension	64	64	64
Number of Layers	2	2 (GRU)	2 (GCN)
GAT Layers	—	2	—
Number of Attention Heads	—	4	—
WaveNet Blocks	—	—	4
Layers per Block	—	—	2
Kernel Size	—	—	2
Output Dimension	64	64	64
Aggregation	mean	mean	mean
Dropout	0.2	0.2	0.2
Bidirectional	false	false	—
Use Edge Attributes	—	false	—
Optimization			
Learning Rate	0.0001	0.0001	0.0001
Weight Decay	0.0001	0.00001	0.0001
Loss Function	MSE	MSE	MSE
Optimizer	Adam	Adam	Adam
Gradient Clipping	1.0	1.0	1.0
Learning Rate Scheduler			
Use Scheduler	true	true	true
Scheduler Type	plateau	plateau	plateau
Scheduler Factor	0.5	0.5	0.5
Scheduler Patience	5	5	5
Scheduler Step Size	20	20	20