TSGFM - Towards a Graph Foundation Model for Time Series Analysis in Network Monitoring

Hamid Latif-Martínez[†‡], Pedro Casas[‡], José Suárez-Varela[§], Albert Cabellos-Aparicio[†], Pere Barlet-Ros[†]
[†]Polytechnic University of Catalonia, Barcelona, Spain
[‡]Austrian Institute of Technology, Vienna, Austria
[§]Telefónica Research, Spain
{hamid.latif, alberto.cabellos, pere.barlet}@upc.edu, jose.suarez-varela@telefonica.com, pedro.casas@ait.ac.at

***Abstract*—We present TSGFM, a Time Series Graph Foundation Model for network monitoring data analysis, based on spatiotemporal Graph Neural Networks (GNN). Inspired by the success of foundation models in achieving generalization and adaptation, TSGFM leverages pretraining on diverse multivariate timeseries (MTS) data from multiple domains to enable effective zero-shot analysis in network monitoring tasks. We compare TSGFM performance against five state-of-the-art AI/ML models in seven zero-shot forecasting scenarios, using five MTS datasets from different domains. Evaluations demonstrate that TSGFM achieves superior performance in six out of seven zero-shot testing scenarios. Most notably, in zero-shot network monitoring analysis, TSGFM surpasses all competing models by at least 18%, even without training on any network monitoring data.**

I. Introduction

Network monitoring is critical to maintain network performance and reliability, with multivariate time series (MTS) analysis enabling key tasks such as anomaly detection and forecasting [1], [2], [3]. Forecasting network behavior is particularly challenging due to traffic complexity and variability [4]. Deep learning methods have become prominent in MTS analysis for their ability to model complex dependencies. Zero-shot learning further enhances these models by allowing generalization to new tasks without retraining. In recent years, foundation models have emerged as an effective means to realize zero-shot learning systems [5]. Their success stems from extensive pretraining on diverse data, which enables them to adapt to new tasks under zero-shot or few-shot conditions.

Inspired by this new paradigm, we introduce the Time Series Graph Foundation Model (TSGFM), a spatio-temporal Graph Neural Network (GNN) designed for zero-shot MTS forecasting. TSGFM captures complex relationships by modeling inter-dependencies among different time series (i.e., spatial correlations) and internal temporal dynamics within each time series (i.e., temporal correlations). To effectively represent these correlations, TSGFM employs an attention-based graph structure, modeling each time series as a node connected through spatial edges in a full-mesh topology, and temporal edges that link nodes across consecutive timesteps.

To assess TSGFM performance, we conduct extensive evaluation using five publicly available datasets spanning multiple diverse domains. We compare TSGFM with a broad selection of state-of-the-art deep learning models used as baseline, including more traditional neural architectures as well as advanced graph-based architectures.

We perform zero-shot testing and cross-validation, training models on a collection of datasets and testing them on an unseen dataset. TSGFM achieves superior performance in six out of seven zero-shot testing scenarios, outperforming all competing models by at least 18% in zero-shot network monitoring, and maintaining a simple and interpretable graph representation.

978-3-903176-74-4

II. Time Series Graph Foundation Model

TSGFM is a GNN-based model that incorporates spatial and temporal information from correlated time series to predict future values of each time series. To address the challenge of spatiotemporal time series forecasting, we propose a foundation model designed to capture complex dependencies across multiple datasets. TSGFM leverages graph-based representations to encode both spatial and temporal correlations between time series. Figure 1 shows an overview TSGFM's architecture and workflow for multivariate time series forecasting. Sliding windowing for context, node embeddings for time-series, spatial and temporal attention, spatiotemporal merge, and readout stages are described next.

A. Spatiotemporal Graph Data Representation

We represent each time series as a node within a graph, enabling the model to exploit the inherent relationships between time series from the same dataset. We construct fully-connected graphs for each dataset, and create spatial edges between each node pair from the same dataset in order to facilitate the propagation of spatial information between time series in the same dataset. When handling multiple datasets, we generate N fully-connected graphs for each time interval t. To capture temporal correlations, we introduce temporal edges that connect each node n at timestep t to itself at future timesteps t', where $t' = t + \Delta t$ and $\Delta t \in \{1, 2, ...\}$, forming edges of the form $(n^t, n^{t'})$, allowing the model to learn the evolution of each time series over time. Inspired by [6], we add a feature Δt to every temporal edge, representing the relative temporal distance between the two nodes.

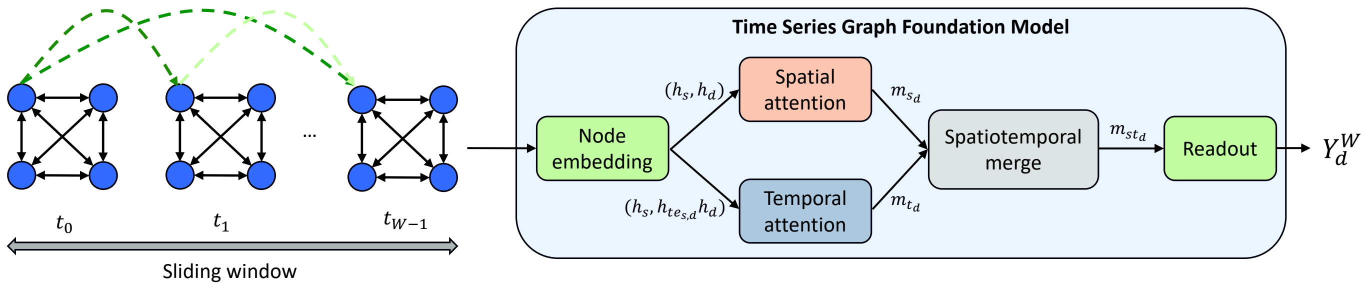

Fig. 1: TSGFM's architecture and workflow for multivariate time series forecasting.

TABLE I: Overview of the publicly available datasets.

Dataset	Description
Spain	Electrical consumption, generation, pricing, and weather data from Spain
ETTh1	Operational data from two electricity substations, including transformer oil levels and temperature readings
Exchange	Daily exchange rates of eight foreign currencies from 1990 to 2016
TELCO	Twelve different time series with a five-minute granularity, representing mobile network measurements
Abilene	Traffic matrix data collected from the Abilene network

We implement a sliding window of size W in the temporal dimension, defining the context of the model. For each timestep t, we generate N independent graphs – one per dataset – where each node contains the value of a specific time series at that time. The sliding window produces a total of $N \times W$ graphs, capturing both the spatial structure at each time step and the temporal progression across the windows.

B. Underlying spatiotemporal GNN model

We implement a Message Passing Neural Network (MPNN) to harness the proposed data representation. We design a message passing composed of two phases executed in parallel: (i) a temporal message passing, and (ii) a spatial message passing. Let a node represent a time series at a specific timestep within the sliding window. We begin by projecting the node features into a hidden representation by using a Multilayer Perceptron (MLP), resulting in an initial hidden state h_n for each node n. This is followed by T message passing iterations, where both spatial and temporal information is propagated. In the temporal message passing phase, we generate an embedding for the temporal edges, denoted as h_{te}, using the Δt attribute, which represents the relative temporal distance between the source and destination nodes. For each temporal edge $te_{s,d}$ connecting the source node s to a destination node d, we assign an attention coefficient $\alpha_{te_{s,d}}$, computed as:

$$\alpha_{te_{s,d}} = MLP(h_s, h_{te_{s,d}}, h_d) \quad (1)$$

$$m_{t_d} = \sum s \in \mathcal{N}_t(d) \alpha_{te_{s,d}} \cdot h_s \quad (2)$$

where h_s and h_d are the embeddings of the source and destination nodes, respectively, and $h_{te_{s,d}}$ is the temporal edge embedding derived from the relative time difference Δt between s and d. The attention coefficients $\alpha_{te_{s,d}}$ are then used to weight the incoming messages received by each node during the aggregation: for each destination node d, we aggregate the incoming messages from its incoming temporal neighbors, $\mathcal{N}_t(d)$, by performing a weighted sum of the source node embedding. Similarly, we model spatial dependencies by assigning attention coefficients to spatial edges between nodes representing different time series at the same timestep. For each spatial edge $se_(s, d)$ we compute:

$$\alpha_{se_{s,d}} = MLP(h_s, h_d) \quad (3)$$

$$m_{s_d} = \sum s \in \mathcal{N}_t(d) \alpha_{te_{s,d}} \cdot h_s \quad (4)$$

Where h_s and h_d are the embeddings of the source and destination nodes, respectively. Rather than updating the node embedding separately with the spatial and temporal messages, m_{s_d} and m_{t_d}, respectively, we first merge these messages in m_{st_d}, and update the node embedding using an $update$ function $h'_d = update(h_d, m_{st_d})$. Finally, after T iterations, we use the hidden states of the nodes from the final timestep in the sliding window (i.e., $t = W - 1$) to produce the forecasting output for each time series at the next timestep using a $readout$ function $Y_d^W = readout(h'_d)$.

III. Evaluation

A. Experimental Setup

1) ***Datasets****:* we evaluate TSGFM using five publicly available datasets spanning various domains, including two datasets from the energy sector [7], [8], a stock-market finance dataset [9] and two datasets from both fixed-line [10] and mobile networks [11]. Table I reports the datasets used in the evaluations. To assess the zero-shot performance and robustness of TSGFM to adapt to new domains, we generate all possible permutations of the datasets using a 4-to-1 split, where four datasets are used for training and one for testing in each permutation. Table II describes the five permutations. The training datasets are further split into 90/10 for training/validation. We apply min-max to normalize the data.

2) ***Baseline models****:* we conduct a comprehensive comparison of TSGFM with a range of representative and state-of-the-art methods to evaluate its performance in MTS forecasting. The baseline models include simpler and more complex

TABLE II: Dataset permutations used for training and testing.

Permutation	**Training**	**Testing**
A	Exchange, **Abilene**, **TELCO**, ETTh1	Spain
B	**TELCO**, Exchange, Spain, ETTh1	**Abilene**
C	Exchange, Spain, **Abilene**, ETTh1	**TELCO**
D	Spain, **Abilene**, **TELCO**, ETTh1	Exchange
E	Exchange, Spain, **Abilene**, **TELCO**	ETTh1

TABLE III: Zero-shot forecasting performance on all five dataset permutations.

Perm. / Model	**A**		**B**		**C**		**D**		**E**		**Avg.**	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
RNN	1.23	1.13	1.62	1.04	0.99	1.00	1.25	1.29	**0.97**	**0.94**	1.52	1.05
FreTS	**0.91**	0.86	1.25	1.01	1.11	1.01	**0.75**	**0.86**	0.99	0.96	1.20	**1.00**
BasicSTGNN	1.32	1.08	2.16	1.12	1.02	0.99	1.75	1.57	0.98	0.95	1.96	1.11
FourierGNN	1.03	0.96	3.29	2.82	**0.98**	**0.98**	**0.75**	**0.86**	1.00	0.98	2.83	2.45
GAT-AD	0.98	**0.82**	1.32	1.22	1.40	1.17	4.75	5.29	2.00	2.12	1.34	1.22
TSGFM	1.00	1.00	**1.00**	**1.00**	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	**1.00**

architectures to study how different complexities and design decisions influence the final results.

We compare against basic spatiotemporal architectures, such as a simple spatiotemporal GNN. This model initializes the hidden state of each node using a RNN, which encodes a sliding window of length W, and updates the node states iteratively through message passing. We also evaluate against GAT-AD [12], a Graph Attention Network (GAT)-based model that incorporates dynamic attention mechanisms to capture spatiotemporal dependencies. We also include FourierGNN [13], a GNN-based architecture that combines frequency domain transformations to efficiently capture spatial and temporal relationships. For non-graph baselines, we include RNNs, which are well-suited for processing sequential data. Finally, we compare with FreTS [14], which leverages frequency-domain transformations and simple MLP architectures.

*3) **Experimental settings**:* all models use the same training, validation and testing datasets, the same split sizes, and the same data normalization strategy. Specifically, we use z-score normalization in all datasets, and we use the mean and standard deviations from each dataset to normalize its data samples. Since we are working in a foundation scenario, where the models are supposed to be trained with vast amounts of data, we cannot assume a training dataset distribution. For this reason, we also normalize the testing samples with the distribution of the testing data set. In all executions, we use a fixed seed $s = 42$ for the sake of reproducibility. All the models are trained using the Mean Absolute Error (MAE) loss function, a learning rate of $\eta = 0.001$, and we set $T = 1$ for TSGFM. We set up early stopping with a patience of 4 epochs based on the training loss.

B. TSGFM vs. Baselines

Table III shows the testing performance obtained by TSGFM and the baseline models for each permutation scenario, in terms of MAE and RMSE. All metrics are normalized with respect to TSGFM, allowing for a direct comparison. The final column reports the average performance of each model, weighted by the number of samples of each dataset.

TSGFM consistently outperforms all competing models when considering the overall weighted performance. Although FourierGNN achieves competitive results in several permutations, its poor performance on the Abilene networking dataset – used in permutation B and the largest dataset in the evaluation – significantly impacts its overall average performance, making it the worst-performing model in aggregate. Across permutations A, C, and E, TSGFM and FourierGNN demonstrate nearly identical performance, indicating that our model is equally effective at capturing spatiotemporal dependencies. However, in permutation B, TSGFM clearly outperforms FourierGNN, highlighting its robustness on larger and more complex datasets. While FourierGNN performs well in some cases, its severe underperformance on the Abilene dataset highlights its lack of robustness.

TABLE IV: Dataset permutations used for training and testing (network scenarios).

Permutation	**Training**	**Testing**
NET-A	Exchange, Spain, ETTh1	TELCO
NET-B	Exchange, Spain, ETTh1	Abilene

TABLE V: Testing performance of forecasting models on network dataset permutations.

Perm. / Model	**NET-A**		**NET-B**		**Avg.**	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
RNN	1.135	1.027	1.686	1.029	1.629	1.029
FreTS	1.096	1.003	1.185	1.016	1.176	1.015
BasicSTGNN	1.474	1.103	3.307	1.293	3.117	1.273
FourierGNN	1.673	1.860	4.832	5.013	4.505	4.687
GAT-AD	1.654	1.243	2.353	1.286	2.281	1.282
TSGFM	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

To further investigate the feasibility of applying foundation models trained on non-network data to the networking domain, we introduce two new evaluation permutations: NET-A and NET-B, as detailed in Table IV. These permutations are specifically designed to evaluate the extent to which learned temporal and structural patterns from non-network datasets can be effectively transferred to network-specific scenarios. The goal is to assess how well TSGFM generalizes to network monitoring tasks without direct exposure to network traffic data during pre-training. Table V reports the performance results on the testing datasets for NET-A and NET-B. Results show that TSGFM consistently outperforms all baseline models across both scenarios, by at least 18% when compared to FreTS, but largely outperforming all the other baselines. TSGFM demonstrates its superior ability to capture meaningful correlations and extract relevant features from diverse data sources. This strong performance suggests that TSGFM effectively transfers learned knowledge from non-network data to network-specific forecasting tasks, reinforcing its potential for network monitoring applications.

Acknowledgment

This work has been supported by the FWF Austrian Science Fund, Project reference I-6653, as well as by the Agencia Estatal de Investigacion, Project reference PCI2023-145974-2, as part of the EU CHIST-ERA-2022-SPiDDS-02 project *GRAPHS4SEC*. This work is also supported by the Catalan Institution for Research and Advanced Studies (ICREA Academia).

References

[1] A. D'Alconzo, I. Drago, A. Morichetta, M. Mellia, and P. Casas, "A survey on big data for network traffic monitoring and analysis," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 800–813, 2019.

[2] P.-W. Tsai, C.-W. Tsai, C.-W. Hsu, and C.-S. Yang, "Network monitoring in software-defined networking: A review," *IEEE Systems Journal*, vol. 12, no. 4, pp. 3958–3969, 2018.

[3] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[4] B. Lim and S. Zohren, "Time-series forecasting with deep learning: a survey," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, p. 20200209, 2021.

[5] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill *et al.*, "On the opportunities and risks of foundation models," *arXiv preprint arXiv:2108.07258*, 2021.

[6] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," *arXiv preprint arXiv:2002.07962*, 2020.

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

[8] N. J. Hana, "Energy consumption, generation, prices and weather dataset," 2021, accessed: 2025-03-28. [Online]. Available: https://www.kaggle.com/datasets/nicholasjhana/energy-consumption-generation-prices-and-weather

[9] G. Lai, "Exchange rate multivariate time series dataset," 2017, accessed: 2025-03-28. [Online]. Available: https://github.com/laiguokun/multivariate-time-series-data/blob/master/exchange_rate/exchange_rate.txt.gz

[10] Y. Zhang, "Abilene traffic matrix dataset," 2004, accessed: 2025-03-28. [Online]. Available: http://cs.utexas.edu/~yzhang/research/AbileneTM/

[11] G. García González, S. Martínez Tagliafico, A. Fernández, G. Gómez, J. Acuña, and P. Casas, "Telco," 2023. [Online]. Available: https://dx.doi.org/10.21227/skpg-0539

[12] H. Latif-Martínez, J. Suárez-Varela, A. Cabellos-Aparicio, and P. Barlet-Ros, "Gat-ad: Graph attention networks for contextual anomaly detection in network monitoring," *Computers & Industrial Engineering*, vol. 200, p. 110830, 2025.

[13] K. Yi, Q. Zhang, W. Fan, H. He, L. Hu, P. Wang, N. An, L. Cao, and Z. Niu, "Fouriergnn: Rethinking multivariate time series forecasting from a pure graph perspective," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[14] K. Yi, Q. Zhang, W. Fan, S. Wang, P. Wang, H. He, N. An, D. Lian, L. Cao, and Z. Niu, "Frequency-domain mlps are more effective learners in time series forecasting," *Advances in Neural Information Processing Systems*, vol. 36, 2024.