DspGNN: Bringing Spectral Design to Discrete Time Dynamic Graph Neural Networks for Edge Regression

Leshanshui Yang
Saagie, LITIS Lab, University of Rouen Normandy
Rouen, FRANCE
leshanshui.yang@saagie.com

Clément Chatelain
LITIS Lab, INSA Rouen Normandy
Rouen, FRANCE
clement.chatelain@insa-rouen.fr

Sébastien Adam
LITIS Lab, University of Rouen Normandy
Rouen, FRANCE
sebastien.adam@univ-rouen.fr

Abstract

We introduce Dynamic Spectral-Parsing Graph Neural Network (DspGNN), a novel model that introduces spectral-designed graph convolution for representation learning and edge regression on Discrete Time Dynamic Graphs (DTDGs). Our first contribution is the optimization of spectral-designed methods for capturing evolving spectral information on DTDGs. In addition, we propose an efficient technique, Active Node Mapping, to address the computational challenge of eigen-decomposition on large DTDGs. Our model consistently outperforms baseline models on three publicly available datasets for edge regression tasks.

1 Introduction and related works

Thanks to the ability of graphs to represent topological structures through nodes and edges, more and more problems in daily life are being modeled using static graphs, extended to dynamic graphs in cases where structures evolve over time [1, 2, 3]. In these networks, nodes represent users/items, and edges reflect temporal relationships with numerical attributes such as transaction volumes [4] and rating scores [5, 6], so accurate prediction is essential for various practical applications. Current research [1, 7, 8] categorizes dynamic graphs into Discrete Time Dynamic Graphs (DTDGs), which represent network states with snapshots, and Continuous Time Dynamic Graphs (CTDGs), which represent network events with timestamped edges. Despite some studies applied on CTDGs [9, 10], there is a scarcity of studies focusing on edge attribute regression on DTDGs, signifying a substantial potential awaiting exploration in this field.

This paper addresses the edge attribute regression problem in DTDGs. We present a novel approach, Dynamic spectral-parsing Graph Neural Network (DspGNN), which is, to the best of our knowledge, the first application of spectral-designed graph convolution networks to DTDGs. We also propose Active Node Mapping, a simple yet efficient technique for eigendecomposition on large sparse DTDGs. Empirical validation on three real-world datasets - Bitcoin-Alpha [11, 12], Bitcoin-OTC [11, 12], and MovieLens [13] - shows not only the efficiency achieved by our approach but also a significant improvement over the baseline models [14, 15], demonstrating its promising potential.

Spectral-Designed Graph Convolution on Static Graphs Graph convolution is a core concept within graph neural networks and falls into two main categories: spatial-based and spectral-based [16], which are explained in detail in appendix A.1. Spatial convolution focuses on local neighbors, aggregating information from reachable neighbors. A spatial convolution layer can be expressed as

Temporal Graph Learning Workshop @ NeurIPS 2023, New Orleans.

$\mathbf{H}' = \sigma\left(\sum_{\mathbf{s}} \mathbf{C}^{(\mathbf{s})}\mathbf{H}\mathbf{W}^{(\mathbf{s})}\right)$ [17], where $\mathbf{H}$ and $\mathbf{H}'$ represent the input and output node hidden state matrices, σ is the activation function, $\mathbf{C}^{(\mathbf{s})}$ represents convolutional kernels (e.g., adjacency matrix $\mathbf{A}$), and $\mathbf{W}$ is the trainable parameter matrix mapping hidden states from dimension $d_{\mathbf{H}}$ to $d_{\mathbf{H}'}$. Spectral convolution filters the signals on the graph in the Fourier domain by the eigendecomposition of the (normalized) Laplacian matrix $\mathbf{L}^{\text{norm}} = \mathbf{D}^{-\frac{\mathbf{1}}{\mathbf{2}}}\mathbf{A}\mathbf{D}^{-\frac{\mathbf{1}}{\mathbf{2}}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathbf{T}}$. A spectral convolution layer can be represented as $\mathbf{H}' = \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^{\mathbf{T}}\mathbf{H}$, where $\mathbf{\Lambda}$ is a diagonal matrix of the eigenvalues, $\mathbf{U}$ is the eigenvector matrix, and $g(\cdot)$ is the filter function(s) that controls the frequency response.

Subsequent research has theoretically analyzed that existing spatial GNNs act as low-pass filters in the spectral domain, and by adding high-pass and/or band-pass filters, their performance can be improved [18]. Building upon this insight, Spectral-designed Graph Convolution (DSGCN) [17] derives convolutional kernels through the eigenvalue filtering of the Laplacian matrix (refer to Eq. 1). Bridging the gap between spectral and spatial graph convolution, DSGCN empirically demonstrates performance beyond spectral-based and spatial-based GNNs.

$$\mathbf{H}' = \sigma\left(\sum_{\mathbf{s}} \mathbf{C}^{(\mathbf{s})}\mathbf{H}\mathbf{W}^{(\mathbf{s})}\right), \quad \mathbf{C}^{\mathbf{s}} = \mathbf{U}\mathbf{\Phi}^{\mathbf{s}}(\mathbf{\Lambda})\mathbf{U}^{\mathbf{T}} \tag{1}$$

Discrete Time Dynamic Graph Neural Networks (DTDGNNs) A DTDG is represented by a sequence of static graphs $(G_1, G_2, ..., G_T)$. Each snapshot G_t is represented by the adjacency matrix $\mathbf{A}_t$, with optional $\mathbf{X}_t^{node}$ or $\mathbf{X}_t^{edge}$ representing node or edge attributes. Tasks on DTDG generally predict attributes or connectivity for the future snapshot(s) based on past K snapshots [14, 15]. Benefiting from the ability to encode each snapshot with static graph encoder [1], early DTDGNNs [19, 20] employ static GNNs f_G to sequentially encode the hidden states for each snapshot and then propagate information across K snapshots using a time module f_T, such as Recurrent Neural Networks (RNNs), as shown in Eq. 2. CoEvoGNN [15], an early contributor to the node regression task on transaction networks, follows this paradigm with residual connected GraphSage as f_G.

$$\mathbf{H'}_k = f_G(\mathbf{A}_k, \mathbf{H}_k), \quad k \in [t-K, t-1], \quad \mathbf{H}_t = f_T(\mathbf{H'}_{t-K:t-1}) \tag{2}$$

Another category of DTDGNNs aims at convolving graph snapshots differently at each time step to better capture graph dynamics [21]. EvolveGCN [14], as a representative example, dynamically updates the GCN weight parameter Θ_{f_G} for each snapshot via a RNN. Despite various research on DTDGNNs, to the best of our knowledge, the architecture design and performance analysis of the spectral-designed method on DTDGs are still largely unexplored.

2 Dynamic Spectral-Parsing Graph Neural Network (DspGNN)

Recent studies have shown that the evolving spectrums of DTDG have some ability to represent dynamic graphs for anomaly detection [22, 23]. Considering the superiority of DSGCN as a static graph encoder, it is intuitive to believe that replacing the graph convolution module in DTDGNN with DSGCN may improve performance. In this section, we delve into this hypothesis and elaborate on how to optimize it for DTDGs. In subsection 2.1, we introduce our model **DspGNN**, which exploits the advantages of the spectral-designed GNN in DTDGNN. Following that, in subsection 2.2, we present an original technique to overcome the challenges of eigendecomposition on DTDGs.

2.1 Spectral-Designed GNN Goes Dynamic

Our model adheres to the common workflow, taking as input the adjacency matrices $\mathbf{A}[t-K, t-1] \in \mathbb{R}^{K\times N\times N}$ and node hidden states $\mathbf{H}[t-K, t-1] \in \mathbb{R}^{K\times N\times d}$ of the past K snapshots. These are processed by DspGNN to encode the hidden states $\mathbf{H} \in \mathbb{R}^{N\times d}$ for the current time step, as shown in Fig. 1 (right). A decision layer then performs the regression task and outputs $\hat{\mathbf{Y}} \in \mathbb{R}^{N\times d_{pred}}$.

In the Spectral Parsing Module, the spectral-designed convolutional kernels are obtained as in DSGCN [17]. In each snapshot, the eigenvalues of $\mathbf{L}^{\text{norm}}$ are filtered by S distinct spectral filters. These filtered spectrums are then inverse-transformed by $\mathbf{U}^{\mathbf{T}}$ to produce various convolutional

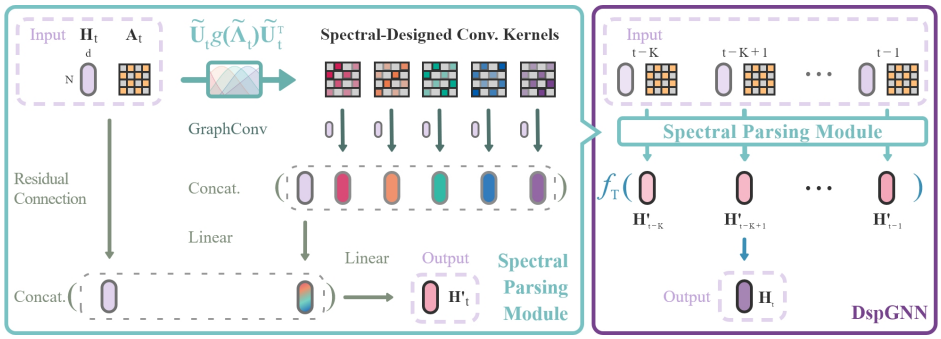

Figure 1: Left: The Spectral Parsing Module, where a set of spectral-designed convolutional kernels are employed to propagate the information. They are then merged with learnable weights to reduce the dimension from $d * 6$ to d. The propagated information is finally concatenated to the input $\mathbf{H}_t$ and passed to a linear layer. Right: The overall architecture of DspGNN involves encoding the hidden states of the past K snapshots by Spectral Parsing Module. This sequence of hidden states is then encoded by a temporal readout function, e.g. LSTM, to obtain the hidden state for the current time step. Predictions can then be computed using a linear layer.

kernels $\mathbf{C}^s = \mathbf{U}g^s(\mathbf{\Lambda})\mathbf{U}^\mathbf{T}$. Each kernel focuses on different spectral frequencies tied to distinct substructures in the graph topology, which can also be thought of as weighted adjacency matrices for spatial convolution. Drawing inspiration from DSGCN [17], our design encompasses a diverse set of filters including a low-pass, a high-pass, and multiple band-pass filters, with center frequency λ_c and bandwidth γ as adjustable parameters. Their frequency responses are shown in appendix A.2.

Our note-worthy improvement to the current DSGCN model involves learning how to combine multiple filters, eliminating the need for manual filter selection across different datasets. While DSGCN manually selects different filters for various datasets, we introduce a linear layer with an activation function to learn the non-linear combination of the outputs from each filter and to reduce the dimensions from $d \times S$ to d. This enables the model to theoretically learn the importance of each filter during the training phase, thus automating the filter selection process.

The final graph encoder architecture, as shown in Fig. 1 (left), not only gains the ability to convolve graph signals from the spectral domain but also automatically combines the information of different spectral domain frequencies. Therefore, we have named it **Spectral Parsing Module**, which acts as the f_G in the overall architecture of DspGNN. A temporal encoder, LSTM, then passes information across multiple snapshots to encode the output hidden states.

2.2 Computing Spectral-Designed Convolutional Kernels on DTDGs

The computational complexity of eigendecomposition poses a major challenge for computing spectral-designed convolutional kernels on Dynamic graphs. The time complexity is $O(N^3)$ per snapshot when using traditional methods where N represents the number of total nodes[1]. To address this issue, we propose a simple yet efficient technique called **Active Node Mapping (ANM)**. This technique maps active nodes in each snapshot to a reduced graph represented by adjacency matrix $\tilde{\mathbf{A}}_t$, shrinking the shape from $N \times N$ to $N_t \times N_t$, where N_t represents the number of nodes which have links at time t (i.e., active nodes of the snapshot), as shown in Fig. 2. ANM significantly improves computational speed and reduces RAM usage, especially when the number of active nodes N_t is low compared to N. The concrete procedure can be found in appendix A.4, theoretical and experimental speedup results are presented in appendix A.5.

[1]Although improved methods for eigendecomposition on sparse matrices exist, such as the Lanczos method [24] which can reduce the time complexity to $O(dN^2)$, where d is the average number of nonzero elements in a row, these approaches still do not address the challenges posed by an exceedingly large number of nodes N.

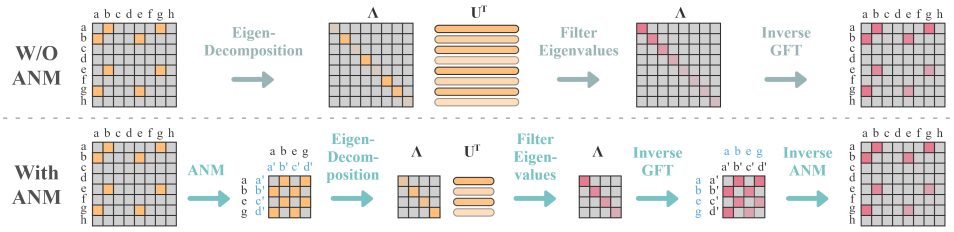

Figure 2: Upper part: eigendecomposition performs on the complete adjacency matrix with N nodes. Lower part: By adding two mapping operations, performed with matrices $\mathbf{M}_t$ and $\mathbf{M^T}_t$, Active Node Mapping applies the eigendecomposition on N_t active nodes instead of total N nodes. Consequently, the theoretical time consumption is reduced to $(\frac{N_t}{N})^3$ of the initial time required.

3 Experiments

Experiments are conducted on three publicly available datasets: Bitcoin-Alpha [11, 12], Bitcoin-OTC [11, 12], and MovieLens-100K [13], see Table 1. The task is to predict the edge attributes in a snapshot, given the adjacency matrices from the previous snapshot(s) as input. In the Bitcoin datasets, which span the interactions of Bitcoin users in anonymous transaction networks, users rate the trust level of others from -10 to +10. In the MovieLens 100K dataset, the edge attributes represent the score with which users rate movies, ranging from 0.5 to 5.0.

Table 1: Datasets

Name	# Nodes	# Average active nodes per snapshot	# Edges	# Time steps	Time step split Train / Valid / Test
Bitcoin-Alpha	3,783	106	24,173	137	95 / 14 / 28
Bitcoin-OTC	5,881	148	35,588	136	95 / 13 / 28
MovieLens-100K	9,811	740	100,836	90	63 / 9 / 18

The mean and standard deviation of the results after the min-max normalization from five runs with different seeds are shown in Table 2. For all three datasets, DspGNN outperforms the baseline models CoEvoGNN and EvolveGCN. Moreover, despite the variability in parameter and node hidden states initialization, DspGNN exhibits much smaller standard deviation margins across the five seeds. These results align with our theoretical expectations, suggesting that our spectral-designed approach can outperform methods based solely on spatial convolution for snapshot encoding.

Table 2: Edge regression performance, showing the mean and standard deviation of the normalized scores for the five seeds, with the best and second best results highlighted in bold and italics.

	Bitcoin-Alpha		Bitcoin-OTC		MovieLens-100K	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
CoEvoGNN*	*0.1171* (0.0097)	*0.0879* (0.0100)	0.1638 (0.0066)	0.1145 (0.0106)	0.2381 (0.0058)	*0.1834* (0.0048)
CoEvoGNN	0.1231 (0.0033)	0.0928 (0.0030)	0.1671 (0.0073)	0.1176 (0.0064)	*0.2372* (0.0059)	0.1894 (0.0067)
EvolveGCN	0.1173 (0.0187)	0.0903 (0.0216)	*0.1556* (0.0049)	*0.1025* (0.0067)	0.2404 (0.0134)	0.1944 (0.0160)
DspGNN	**0.0968** (0.0002)	**0.0637** (0.0006)	**0.1471** (0.0009)	**0.0831** (0.0019)	**0.2293** (0.0000)	**0.1812** (0.0000)

4 Conclusion

In this work, we presented DspGNN, a model based on static graph spectral-designed convolution [17], and inspired by both CoEvoGNN's architecture for dynamic graph regression [15] and EvolveGCN's adaptive graph convolution [14]. Our approach offers three main novelties: 1) it adapts spectral-designed methods for enhanced compatibility with DTDGs 2) it improves upon the existing DSGCN model by combining the information of different kernels with a learnable linear layer, and 3) it innovatively solves the challenge of eigendecomposition on DTDGs with Active Node Mapping.

Acknowledgment This work was financially supported by the ANR Labcom Lisa ANR-20-LCV1-0009.

References

[1] Joakim Skarding, Bogdan Gabrys, and Katarzyna Musial. Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey. *IEEE Access*, 9:79143–79168, 2021.

[2] Yue Zhou, Xin Luo, and MengChu Zhou. Cryptocurrency transaction network embedding from static and dynamic perspectives: An overview. *IEEE/CAA Journal of Automatica Sinica*, 10(5):1105–1121, 2023.

[3] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[4] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. *arXiv preprint arXiv:1908.02591*, 2019.

[5] Gideon Dror, Noam Koenigstein, Yehuda Koren, and Markus Weimer. The yahoo! music dataset and kdd-cup'11. In *Proceedings of KDD Cup 2011*, pages 3–18. PMLR, 2012.

[6] Ioannis Konstas, Vassilios Stathopoulos, and Joemon M Jose. On social networks and collaborative recommendation. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 195–202. ACM, 2009.

[7] Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation learning for dynamic graphs: A survey. *The Journal of Machine Learning Research*, 21(1):2648–2720, 2020.

[8] Antonio Longa, Veronica Lachi, Gabriele Santin, Monica Bianchini, Bruno Lepri, Pietro Lio, Franco Scarselli, and Andrea Passerini. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *arXiv preprint arXiv:2302.01018*, 2023.

[9] Mohit Raghavendra, Kartik Sharma, Srijan Kumar, et al. Signed link representation in continuous-time dynamic signed networks. *arXiv preprint arXiv:2207.03408*, 2022.

[10] Lekang Jiang, Caiqi Zhang, Farimah Poursafaei, and Shenyang Huang. Towards temporal edge regression: A case study on agriculture trade between nations. *arXiv preprint arXiv:2308.07883*, 2023.

[11] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 221–230. IEEE, 2016.

[12] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 333–341. ACM, 2018.

[13] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[14] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5363–5370, 2020.

[15] Daheng Wang, Zhihan Zhang, Yihong Ma, Tong Zhao, Tianwen Jiang, Nitesh Chawla, and Meng Jiang. Modeling co-evolution of attributed and structural information in graph sequence. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[16] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[17] Muhammet Balcilar, Guillaume Renton, Pierre Héroux, Benoit Gauzere, Sebastien Adam, and Paul Honeine. Bridging the gap between spectral and spatial domains in graph neural networks. *arXiv preprint arXiv:2003.11702*, 2020.

[18] Muhammet Balcilar, Renton Guillaume, Pierre Héroux, Benoit Gaüzère, Sébastien Adam, and Paul Honeine. Analyzing the expressive power of graph neural networks in a spectral perspective. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[19] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25*, pages 362–373. Springer, 2018.

[20] Franco Manessi, Alessandro Rozza, and Mario Manzo. Dynamic graph convolutional networks. *Pattern Recognition*, 97:107000, 2020.

[21] Leshanshui Yang, Sébastien Adam, and Clément Chatelain. Dynamic graph representation learning with neural networks: A survey. *arXiv preprint arXiv:2304.05729*, 2023.

[22] Shenyang Huang, Yasmeen Hitti, Guillaume Rabusseau, and Reihaneh Rabbany. Laplacian change point detection for dynamic graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 349–358, 2020.

[23] Shenyang Huang, Jacob Danovitch, Guillaume Rabusseau, and Reihaneh Rabbany. Fast and attributed change detection on dynamic graphs with density of states. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 15–26. Springer, 2023.

[24] Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[25] Alessio Micheli. Neural network for graphs: A contextual constructive approach. *IEEE Transactions on Neural Networks*, 20(3):498–511, 2009.

[26] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[27] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[28] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

[29] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[30] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.