Behavioral-Similarity and Clustering-Based Methods for Static Graph Estimation in Hybrid GNNs (Student Abstract)

Ryusei Otani[1], **Keiichi Namikoshi**[1], **Yuko Sakurai**[1], **Mingyu Guo**[2], **Satoshi Oyama**[3]
[1] Nagoya Institute of Technology,
[2] University of Adelaide,
[3] Nagoya City University/ RIKEN AIP
{r.otani.638@stn., namikoshi.keiichi@, sakurai@}nitech.ac.jp, mingyu.guo@adelaide.edu.au, oyama@ds.nagoya-cu.ac.jp

Abstract

In this study, we propose two methods to estimate static graphs from a single dynamic graph and integrate them into hybrid Graph Neural Networks (GNNs), which combine long-term static structure with transient dynamic interactions. Since static graphs are often unavailable and attributes may be difficult to use at scale or under privacy constraints, we introduce: (i) a "behavioral similarity" estimator based on normalized co-occurrence, which requires no attributes, and (ii) an attribute-aware K-means + k-NN estimator that is more efficient than cosine similarity. Experiments on multiple real-world datasets show that both methods consistently improve predictive accuracy and training efficiency, underscoring the importance of static graph choice in hybrid GNNs.

Introduction

Graph Neural Networks (GNNs) have shown strong performance in domains such as social networks, transportation, finance, and molecular analysis. Dynamic GNNs (DGNNs), which model temporally evolving graphs, have also advanced rapidly (Pang, Jia, and Deng 2022). More recently, hybrid GNNs that combine static structures with dynamic relations have gained attention: the static branch captures long-term dependencies, while the dynamic branch models short-term interactions, boosting predictive accuracy (Yuan et al. 2025). In practice, however, static graphs are often unavailable due to privacy or access restrictions, making it necessary to extract usable static structures from a single dynamic graph. Existing methods typically rely on attribute-based similarity (Lin, Chen, and Wang 2022); however, these often introduce noisy edges, reducing accuracy and efficiency in hybrid GNNs.

To address this issue, we propose two methods: (1) Co-occurrence, which aggregates and normalizes node co-occurrence frequencies over time windows to connect nodes with similar behavioral patterns without requiring attributes, and (2) Cluster, which applies K-means clustering to node attributes and then builds sparse, high-quality static graphs by connecting nodes within each cluster via k-NN. The former reflects "behavioral similarity" to improve predictive accuracy, while the latter estimates efficient static structures that enhance training efficiency. Experiments on real-world datasets demonstrate that the co-occurrence estimator yields significant improvements in predictive accuracy, whereas the cluster estimator achieves clear gains in training efficiency. These results highlight the importance of selecting appropriate static graphs in hybrid GNNs, an aspect that has received little attention so far.

Proposed Method

Overview. Figure 1 shows our framework: from a single dynamic graph, we build a static graph, encode static and dynamic branches separately, and fuse their embeddings with learnable weights for node classification.

Co-occurrence. Given node set V, contexts $\mathcal{C} = \{S_c\}$, threshold τ, and maximum degree k, we build the static graph $G = (V, E)$ as follows:

1. Compute co-occurrence counts:
$$\mathrm{cooc}(u, v) \;=\; \sum_{c \in \mathcal{C}} \mathbf{1}\{\, u \in S_c \wedge v \in S_c \,\}.$$
2. For each node u, select top-k neighbors v with $\mathrm{cooc}(u, v) \geq \tau$ and add edges $\{u, v\}$ to E.

By linking nodes with high co-occurrence in shared contexts, the constructed graph reflects behavioral similarity.

Cluster. Given node set V, features $X = \{x_i\}$, cluster number n_c, and neighbor size k_{nn}:

1. Apply K-means clustering to X:
$$\min_{c(\cdot), \{\mu_j\}} \sum_{i=1}^{N} \|x_i - \mu_{c(i)}\|_2^2.$$
where $c(i)$ is the cluster assignment of x_i.
2. Within each cluster C_j, connect each $u \in C_j$ to its top-k_{nn} most similar nodes by cosine similarity

This procedure first groups nodes by attributes and then connects neighbors within each group, resulting in a sparse user–similarity graph.

Model Integration. A static encoder f_S and a dynamic encoder f_D output embeddings $H^{(s)}, H^{(d)} \in \mathbf{R}^{N \times F}$. Per-node learnable weights $\mathbf{w}_v^{(s)}, \mathbf{w}_v^{(d)}$ (softmax-normalized) balance static and dynamic signals:
$$\mathbf{h}_v^{(f)} = \mathbf{w}_v^{(s)} \odot \mathbf{h}_v^{(s)} + \mathbf{w}_v^{(d)} \odot \mathbf{h}_v^{(d)}.$$
Given the fused embeddings $\mathbf{h}_v^{(f)}$, a shallow MLP then outputs a logit ℓ_v and a probability $\hat{p}_v = \sigma(\ell_v)$.

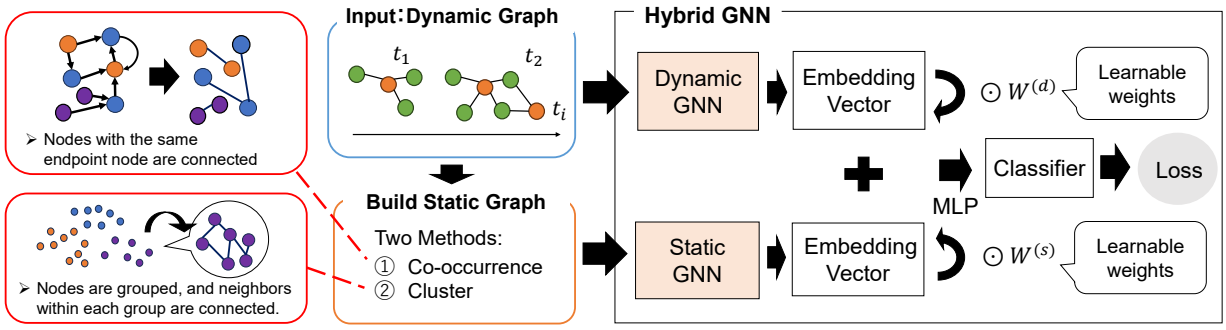

Figure 1: Framework: Two static graph builders and a hybrid GNN.

Dataset	Type	Nodes	Edges	Feat.	Cls.
MOOC	bipartite	7,144	411,749	4	2
Wikipedia	bipartite	9,227	157,474	172	2
Reddit	bipartite	10,984	672,447	172	2

Table 1: Dataset statistics

	Cosine		Cooc (our)		Cluster (our)	
Dataset	AUC	$t(s)$	AUC	$t(s)$	AUC	$t(s)$
MOOC	0.9673	65.9	**0.9724**	16.3	0.9675	**7.2**
Wikipedia	0.9641	25.3	**0.9673**	10.0	0.9648	**7.3**
Reddit	0.8340	23.0	**0.8402**	17.4	0.8324	**13.1**

Table 2: Experimental results of node classification

Hardware: Intel Core i9-12900K (16c/24t, 3.2 GHz), 128 GB RAM; NVIDIA GeForce RTX 3070 Ti (8 GB).
Note: Standard deviations over 10 seeds are small (MOOC $\leq$ .001, Wikipedia $\leq$ .005, Reddit $\leq$ .011).

Experimental Setting

We evaluate *Cooc* and *Cluster* against a cosine baseline on three temporal datasets—MOOC, Wikipedia, and Reddit from JODIE (Kumar, Zhang, and Leskovec 2019) (Table 1)—for node classification.

Static-graph builders.

- *Cosine (baseline):* add edges for pairs with cosine ≥ 0.8.
- *Cooc (ours):* Co-occurrence with τ=1, k=100.
- *Cluster (ours):* Cluster with n_c=5, k_{nn}=5.

Model and training. We train a hybrid GNN composed of GraphSAGE (Hamilton, Ying, and Leskovec 2017) for the estimated static graphs and T-GCN (Zhao et al. 2020) for dynamic graphs, with a hidden size of 64, a dropout rate of 0.5, the Adam optimizer, and Binary Cross-Entropy with `pos_weight`. To avoid temporal leakage, the data are split chronologically into 8/1/1 (train/validation/test) by event time with early stopping on the validation set. We report the mean AUC over 10 seeds as the classification metric and the training runtime $t(s)$ as an indicator of learning efficiency.

Experimental Results

Across all datasets, *Cooc* achieved the highest AUC, confirming that behavior co-occurrence–based structure is effective for improving predictive accuracy. In contrast, *Cluster* consistently attained the shortest training time and exhibited higher computational efficiency than *Cosine*. Therefore, *Cooc* is reasonable when accuracy is prioritized, whereas *Cluster* is preferable when efficiency is prioritized.

The advantage of *Cooc* is that it directly encodes, as a static structure, similarity in behavior that is difficult to capture using attributes alone. For example, in a MOOC, nodes represent students and courses, edges represent enrollment histories, and the task is dropout prediction. Co-occurrence connects students who tend to choose the same courses, embedding latent factors—such as interests, learning tendencies, and course difficulty—into the static graph. This complements the dynamic branch and leads to greater discriminative power than the attribute-based approach (*Cosine*).

By contrast, *Cluster* first forms coarse-grained groupings in the attribute space and then links local neighbors, yielding a sparser graph and lighter training. Although its accuracy is slightly below that of *Cooc*, it substantially reduces training time, making it useful as a basis for hyperparameter search and continual training. Overall, the choice of the static-graph construction method has a significant impact on both the accuracy and the training efficiency of hybrid models.

Conclusion

We proposed two static-graph construction methods to improve hybrid GNN performance. Experiments showed that *Cooc* performs well in accuracy, while *Cluster* performs well in training efficiency, underscoring that the choice of static structure governs both. As future work, a key direction is to jointly optimize the static graph in sync with training so that it converges to a sparser, more informative structure.

Acknowledgements

This research was partially supported by JSPS KAKENHI (Grant Number 24K01112) and by JST CREST (Grant Numbers JPMJCR21D1 and JPMJCR2564).

References

Hamilton, W. L.; Ying, R.; and Leskovec, J. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 1025–1035.

Kumar, S.; Zhang, X.; and Leskovec, J. 2019. Predicting Dynamic Embedding Trajectory in Temporal Interaction Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1269–1278.

Lin, J.; Chen, S.; and Wang, J. 2022. Graph Neural Networks with Dynamic and Static Representations for Social Recommendation. In *Database Systems for Advanced Applications*, 264–271. Cham: Springer International Publishing. ISBN 978-3-031-00126-0.

Pang, J.; Jia, L.; and Deng, J. 2022. A Survey on Dynamic Graph Neural Networks Modeling. In *2022 China Automation Congress (CAC)*, 2476–2481.

Yuan, Q.; Liu, Y.; Tang, Y.; Chen, X.; Zheng, X.; He, Q.; and Ao, X. 2025. Dynamic Graph Learning with Static Relations for Credit Risk Assessment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 13133–13141.

Zhao, L.; Song, Y.; Zhang, C.; Liu, Y.; Wang, P.; Lin, T.; Deng, M.; and Li, H. 2020. T-GCN: A Temporal Graph Convolutional Network for Traffic Prediction. *IEEE Transactions on Intelligent Transportation Systems*, 21(9): 3848–3858.