References

Andrew Anderson, Jonathan Dodge, Amrita Sadarangani, Zoe Juozapaitis, Evan Newman, Jed Irvine, Souti Chattopadhyay, Matthew Olson, Alan Fern, and Margaret Burnett. Mental models of mere mortals with explanations of reinforcement learning. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 10(2):1–37, 2020.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

Federico Baldassarre and Hossein Azizpour. Explainability techniques for graph convolutional networks. In *ICML Workshop*, 2019.

Asim Kumar Debnath, Rosa L. Lopez de Compadre, Gargi Debnath, Alan J. Shusterman, and Corwin Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. 1991.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*.

Kaitlyn M Gayvert, Neel S Madhukar, and Olivier Elemento. A data-driven approach to predicting successes and failures of clinical trials. *Cell chemical biology*, 23(10):1294–1301, 2016.

Shurui Gui, Hao Yuan, Jie Wang, Qicheng Lao, Kang Li, and Shuiwang Ji. Flowx: Towards explainable graph neural networks via message flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. *arXiv preprint arXiv:2001.06216*, 2020.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Vivian Lai and Chenhao Tan. On Human Predictions with Explanations and Predictions of Machine Learning Models: A Case Study on Deception Detection. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2019.

Yunqi Li, Yingqiang Ge, and Yongfeng Zhang. Tutorial on fairness of machine learning in recommender systems. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pp. 2654–2657, 2021.

Yazheng Liu, Xi Zhang, and Sihong Xie. A differential geometric view and explainability of gnn on evolving graphs. *arXiv preprint arXiv:2403.06425*, 2024.

Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.

Jing Ma, Wei Gao, and Kam-Fai Wong. Rumor detection on twitter with tree-structured recursive neural networks. ACL, 2018.

Phillip E Pope, Soheil Kolouri, Mohammad Rostami, Charles E Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *CVPR*, 2019.

Gabriëlle Ras, Marcel van Gerven, and Pim Haselager. Explanation methods in deep learning: Users, values, concerns and challenges. *Explainable and interpretable models in computer vision and machine learning*, pp. 19–36, 2018.

Shebuti Rayana and Leman Akoglu. Collective opinion spam detection: Bridging review networks and metadata. In *KDD*, 2015.

Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. Interpreting graph neural networks for nlp with differentiable edge masking. *arXiv preprint arXiv:2010.00577*, 2020.

Thomas Schnake, Oliver Eberle, Jonas Lederer, Shinichi Nakajima Kristof T. Sch¨utt, Klaus-Robert M¨uller, and Gr´egoire Montavon. Higher-order explanations of graph neural networks via relevant walks. 2020.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pp. 3145–3153. PMlR, 2017.

Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *KDD*, 2019a.

Jianyu Wang, Rui Wen, Chunming Wu, Yu Huang, and Jian Xion. FdGars: Fraudster Detection via Graph Convolutional Networks in Online App Review System. In *WWW*, 2019b.

Zhenqin Wu, Bharath Ramsundarand Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay S. Pande. Moleculenet: a benchmark for molecular machine learning. 2018.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*, 2018.

Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. GNNExplainer: Generating Explanations for Graph Neural Networks. In *NeurIPS*, 2019.

Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. Xgnn: Towards model-level explanations of graph neural networks. 2020a.

Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *arXiv preprint arXiv:2012.15445*, 2020b.

Arkaitz Zubiaga, Maria Liakata, and Rob Procter. Exploiting context for rumour detection in social media. In *ICSI*, pp. 109–123, 2017.

A APPENDIX

A.1 EVALUATION OF MESSAGE FLOWS ON DYNAMIC GRAPHS

In Figure 6, we illustrate the computation of Fidelity for both dynamic and static graphs from the perspective of computational graphs. The static graph G_0 is considered an evolution of G_{empty}. In the case of dynamic graphs, G_1 evolves from the G_0. After identifying the important message flows, we adjust their weights to align with those in the destination graph, keeping the weights of the remaining flows unchanged. This process generates a new computational graph G_n. In dynamic graphs, adjusting the weights of selected important message flows may lead to differing weights for the same-layer edges across various flows. However, GNN propagation rules require that edges within each layer share a single weight. Thus, merging these flows while complying with GNN propagation constraints is infeasible.

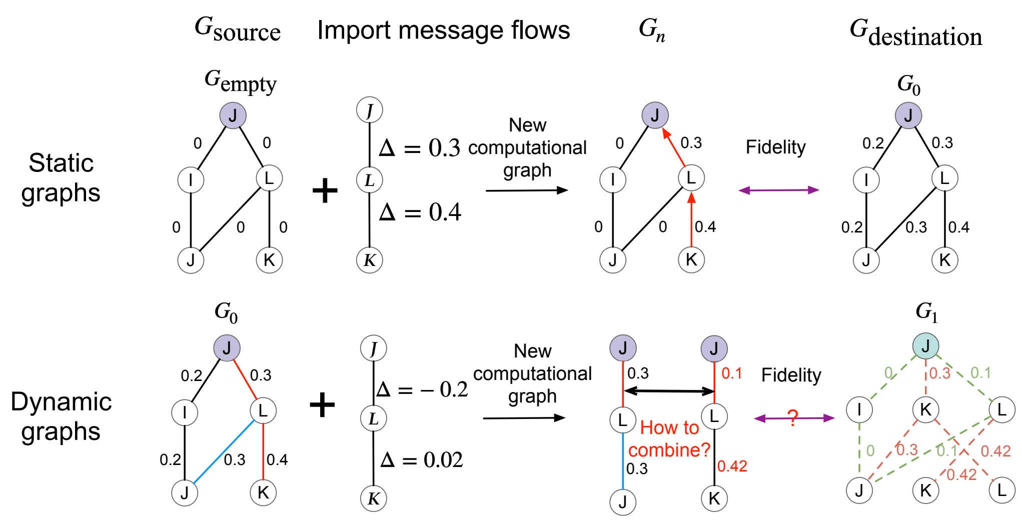

Figure 6: Calculation of Fidelity for dynamic and static graphs. Challenges may arise during the computation for dynamic graphs.

A.2 CALCULATE THE CONTRIBUTION OF MESSAGE FLOWS

A.2.1 THE EXAMPLES ON THE NODE PREDICTION TASKS

Supposing the GNN models have two layers, considering the massage flow $\mathcal{F} = (V, I, J) \in$ the altered message flows set $\Delta\mathcal{F}$, We have derived in detail the calculation process of the contribution value of message flow:

$$
\begin{aligned}
\mathbf{C}_s &= a_{IJ}^{0,T}\Delta\mathbf{h}_I^{t-1}\boldsymbol{\theta^T} + \Delta a_{IJ}^{t}\mathbf{h}_I^{1,t-1}\boldsymbol{\theta^T} \quad \text{the contribution of } \Delta\mathbf{h}_I, \mathbf{h}_I \text{ to } \Delta\mathbf{z}_J \\
&= a_{IJ}^{0,T}\left(\Delta\mathbf{z}_I^{T-1}\mathbf{m}_{\Delta\mathbf{z}_I^{T-1}\Delta\mathbf{h}_I^{T-1}}\right)\boldsymbol{\theta^T} \quad \text{the contribution of } \Delta\mathbf{z}_I \text{ to } \Delta\mathbf{h}_I \\
&\quad + \Delta a_{IJ}^{T}\left(\mathbf{z}_I^{1,T-1}\mathbf{m}_{\mathbf{z}_I^{1,T-1}\mathbf{h}_I^{1,T-1}}\right)\boldsymbol{\theta^T} \quad \text{the contribution of } \mathbf{z}_I \text{ to } \mathbf{h}_I \\
&= a_{IJ}^{0,T}\Delta\mathbf{h}_V^{T-2}\mathbf{m}_{\Delta\mathbf{h}_V^{T-2}\Delta\mathbf{z}_I^{T-1}}\mathbf{m}_{\Delta\mathbf{z}_I^{T-1}\Delta\mathbf{h}_I^{T-1}}\boldsymbol{\theta^T} \quad \text{the contribution of } \Delta\mathbf{h}_V \text{ to } \Delta\mathbf{z}_J \\
&\quad + a_{IJ}^{0,T}\mathbf{h}_V^{1,T-2}\mathbf{m}_{\mathbf{h}_V^{1,T-2}\Delta\mathbf{z}_I^{T-1}}\mathbf{m}_{\Delta\mathbf{z}_I^{T-1}\Delta\mathbf{h}_I^{T-1}}\boldsymbol{\theta^T} \quad \text{the contribution of } \mathbf{h}_V \text{ to } \Delta\mathbf{z}_J \\
&\quad + \Delta a_{IJ}^{T}\left(\mathbf{h}_V^{1,T-2}\mathbf{m}_{\mathbf{h}_V^{1,T-2}\mathbf{z}_I^{1,T-1}}\right)\mathbf{m}_{\mathbf{z}_I^{1,T-1}\mathbf{h}_I^{1,T-1}}\boldsymbol{\theta^T} \quad \text{the contribution of } \mathbf{h}_V \text{ to } \Delta\mathbf{z}_J
\end{aligned} \tag{10}
$$

According to the multiplier designed by the DeepLIFT, $\mathbf{m}_{\Delta\mathbf{h}_V^{T-2}\Delta\mathbf{z}_I^{T-1}} = \Delta a_{VI}^{T-1}\boldsymbol{\theta^{T-1}}$, $\mathbf{m}_{\Delta\mathbf{z}_I^{T-1}\Delta\mathbf{h}_I^{T-1}} = \frac{\Delta\mathbf{h}_I^{T-1}}{\Delta\mathbf{z}_I^{T-1}}$, $\mathbf{m}_{\mathbf{z}_I^{T-1}\mathbf{h}_I^{T-1}} = \frac{\mathbf{h}_I^{T-1}}{\mathbf{z}_I^{T-1}}$, $\mathbf{m}_{\mathbf{h}_V^{1,T-2}\mathbf{z}_I^{1,T-1}} = a_{VI}^{1,T-1}\boldsymbol{\theta^{T-1}}$, therefore,

$$\mathbf{C}_s = \Delta a_{VI}^{T-1} a_{IJ}^{0,T} \mathbf{h}_V^{1,T-2} \boldsymbol{\theta^{T-1}} \frac{\Delta \mathbf{h}_I^{T-1}}{\Delta \mathbf{z}_I^{T-1}} \boldsymbol{\theta^{T}} + a_{VI}^{1,T-1} \Delta a_{IJ}^{T} \mathbf{h}_V^{1,T-2} \boldsymbol{\theta^{T-1}} \frac{\mathbf{h}_I^{T-1}}{\mathbf{z}_I^{T-1}} \boldsymbol{\theta^{T}} \tag{11}$$

Where the divide means the element-wise division, $T = 2$.

Similarly, Supposing the GNN models have three layers, considering the massage flow $\mathcal{F} = (U, V, I, J) \in$ the altered message flows set $\Delta\boldsymbol{\mathcal{F}}$, We have derived in detail the calculation process of the contribution value of message flow:

$$\begin{aligned}
\mathbf{C}_s &= a_{IJ}^{0,T} \Delta \mathbf{h}_I^{t-1} \boldsymbol{\theta^{T}} + \Delta a_{IJ}^{t} \mathbf{h}_I^{1,t-1} \boldsymbol{\theta^{T}} \quad \text{the contribution of } \Delta \mathbf{h}_I, \mathbf{h}_I \text{ to } \Delta \mathbf{z}_J \\
&= a_{IJ}^{0,T} \left(\Delta \mathbf{z}_I^{T-1} \mathbf{m}_{\Delta \mathbf{z}_I^{T-1} \Delta \mathbf{h}_I^{T-1}}\right) \boldsymbol{\theta^{T}} \quad \text{the contribution of } \Delta \mathbf{z}_I \text{ to } \Delta \mathbf{h}_I \\
&\quad + \Delta a_{IJ}^{T} \left(\mathbf{z}_I^{1,T-1} \mathbf{m}_{\mathbf{z}_I^{1,T-1} \mathbf{h}_I^{1,T-1}}\right) \boldsymbol{\theta^{T}} \quad \text{the contribution of } \mathbf{z}_I \text{ to } \mathbf{h}_I \\
&= a_{IJ}^{0,T} \Delta \mathbf{h}_V^{T-2} \mathbf{m}_{\Delta \mathbf{h}_V^{T-2} \Delta \mathbf{z}_I^{T-1}} \mathbf{m}_{\Delta \mathbf{z}_I^{T-1} \Delta \mathbf{h}_I^{T-1}} \boldsymbol{\theta^{T}} \quad \text{the contribution of } \Delta \mathbf{h}_V \text{ to } \Delta \mathbf{z}_J \\
&\quad + a_{IJ}^{0,T} \mathbf{h}_V^{1,T-2} \mathbf{m}_{\mathbf{h}_V^{1,T-2} \Delta \mathbf{z}_I^{T-1}} \mathbf{m}_{\Delta \mathbf{z}_I^{T-1} \Delta \mathbf{h}_I^{T-1}} \boldsymbol{\theta^{T}} \quad \text{the contribution of } \mathbf{h}_V \text{ to } \Delta \mathbf{z}_J \\
&\quad + \Delta a_{IJ}^{T} \left(\mathbf{h}_V^{1,T-2} \mathbf{m}_{\mathbf{h}_V^{1,T-2} \mathbf{z}_I^{1,T-1}}\right) \mathbf{m}_{\mathbf{z}_I^{1,T-1} \mathbf{h}_I^{1,T-1}} \boldsymbol{\theta^{T}} \quad \text{the contribution of } \mathbf{h}_V \text{ to } \Delta \mathbf{z}_J \\
&= a_{IJ}^{0,T} \left(a_{UV}^{0,T-2} \Delta \mathbf{h}_U^{T-3} \boldsymbol{\theta^{T-2}} + \Delta a_{UV}^{T-2} \mathbf{h}_U^{1,T-3} \boldsymbol{\theta^{T-2}}\right) \mathbf{m}_{\Delta \mathbf{h}_V^{T-2} \Delta \mathbf{z}_I^{T-1}} \mathbf{m}_{\Delta \mathbf{z}_I^{T-1} \Delta \mathbf{h}_I^{T-1}} \boldsymbol{\theta^{T}} \\
&\quad \text{the contribution of } \Delta \mathbf{h}_U \text{ to } \Delta \mathbf{z}_J \\
&\quad + a_{IJ}^{0,T} \left(\mathbf{h}_U^{T-3} \mathbf{m}_{\mathbf{h}_U^{1,T-3} \mathbf{z}_V^{1,T-2}} \mathbf{m}_{\mathbf{z}_V^{1,T-2} \mathbf{h}_V^{1,T-2}}\right) \mathbf{m}_{\mathbf{h}_V^{1,T-2} \Delta \mathbf{z}_I^{T-1}} \mathbf{m}_{\Delta \mathbf{z}_I^{T-1} \Delta \mathbf{h}_I^{T-1}} \boldsymbol{\theta^{T}} \\
&\quad \text{the contribution of } \mathbf{h}_U \text{ to } \Delta \mathbf{z}_J \\
&\quad + \Delta a_{IJ}^{T} \left(\mathbf{h}_U^{T-3} \mathbf{m}_{\mathbf{h}_U^{1,T-3} \mathbf{z}_V^{1,T-2}} \mathbf{m}_{\mathbf{z}_V^{1,T-2} \mathbf{h}_V^{1,T-2}}\right) \mathbf{m}_{\mathbf{h}_V^{1,T-2} \mathbf{z}_I^{1,T-1}} \mathbf{m}_{\mathbf{z}_I^{1,T-1} \mathbf{h}_I^{1,T-1}} \boldsymbol{\theta^{T}} \\
&\qquad \text{the contribution of } \mathbf{h}_U \text{ to } \Delta \mathbf{z}_J \\
&= \Delta a_{UV}^{0,T-2} a_{VI}^{0,T-1} a_{IJ}^{0,T} \mathbf{h}_U^{1,T-3} \boldsymbol{\theta^{T-2}} \frac{\Delta \mathbf{h}_V^{T-2}}{\Delta \mathbf{z}_V^{T-2}} \boldsymbol{\theta^{T-1}} \frac{\Delta \mathbf{h}_I^{T-1}}{\Delta \mathbf{z}_I^{T-1}} \boldsymbol{\theta^{T}} \\
&\quad + a_{UV}^{1,T-2} \Delta a_{VI}^{T-1} a_{IJ}^{0,T} \mathbf{h}_U^{1,T-3} \boldsymbol{\theta^{T-2}} \frac{\mathbf{h}_V^{T-2}}{\mathbf{z}_V^{T-2}} \boldsymbol{\theta^{T-1}} \frac{\Delta \mathbf{h}_I^{T-1}}{\Delta \mathbf{z}_I^{T-1}} \boldsymbol{\theta^{T}} \\
&\quad + a_{UV}^{1,T-2} a_{VI}^{1,T-1} \Delta a_{IJ}^{T} \mathbf{h}_U^{1,T-3} \boldsymbol{\theta^{T-2}} \frac{\mathbf{h}_V^{T-2}}{\mathbf{z}_V^{T-2}} \boldsymbol{\theta^{T-1}} \frac{\mathbf{h}_I^{T-1}}{\mathbf{z}_I^{T-1}} \boldsymbol{\theta^{T}}
\end{aligned} \tag{12}$$

A.2.2 On the link prediction task

According to the equation 3, for the target edge e_{IJ}, the $\mathbf{z}_I^T \in \mathbb{R}^{1\times d}$ and $\mathbf{z}_J^T \in \mathbb{R}^{1\times d}$ are concatenated, and fed into a linear layer with the parameters $\boldsymbol{\theta_{LP}}$. According to the equation 7, we can obtain the contribution of message flow $\mathcal{F}_{V_1, V_2, \cdots, V_T, V_{T+1}}$ to $\Delta \mathbf{z}_I^T$ or $\Delta \mathbf{z}_J^T$, then the contribution of message flow to the $\Delta \mathbf{z}_{IJ} = \mathbf{z}_{IJ}(G_1) - \mathbf{z}_{IJ}(G_0)$ is:

$$\begin{aligned}
\mathbf{C_s} = \sum_{t=0}^{T-1} \Bigg(& a_{\mathcal{F}[0]\mathcal{F}[1]}^{1,1} a_{\mathcal{F}[1]\mathcal{F}[2]}^{1,2} \cdots \Delta a_{\mathcal{F}[t]\mathcal{F}[t+1]}^{t+1} a_{\mathcal{F}[t+1],\mathcal{F}[t+2]}^{0,t+2} \cdots a_{\mathcal{F}[T-1],\mathcal{F}[T]}^{0,T} \\
& \mathbf{h}_{\mathcal{F}[0]}^{1,0} \frac{\mathbf{h}_{\mathcal{F}[1]}^{1,1}}{\mathbf{z}_{\mathcal{F}[1]}^{1,1}} \cdots \frac{\mathbf{h}_{\mathcal{F}[t]}^{1,t}}{\mathbf{z}_{\mathcal{F}[t]}^{1,t}} \boldsymbol{\theta}^{t} \frac{\Delta \mathbf{h}_{\mathcal{F}[t+1]}^{t+1}}{\Delta \mathbf{z}_{\mathcal{F}[t+1]}^{t+1}} \boldsymbol{\theta}^{t+1} \cdots \frac{\Delta \mathbf{h}_{\mathcal{F}[T-1]}^{T-1}}{\Delta \mathbf{z}_{\mathcal{F}[T-1]}^{T-1}} \boldsymbol{\theta}^{T} \boldsymbol{\theta}'_{LP} \Bigg)
\end{aligned} \tag{13}$$

Where $\boldsymbol{\theta}'_{LP} = \boldsymbol{\theta}_{LP}[0:d]$, d if $V_{T+1} = I$, $\boldsymbol{\theta}'_{LP} = \boldsymbol{\theta}_{LP}[d:]$, if $V_{T+1} = J$

A.2.3 On the graph classification task

Because the average pooling is used for the graph classification tasks, $\Delta\mathbf{z} = \mathbf{z}(G_1) - \mathbf{z}(G_0) = \sum_{J\in\left(\mathcal{V}^0\cup\mathcal{V}^1\right)} \Delta\mathbf{z}_J^T / |\mathcal{V}^0 \cup \mathcal{V}^1|$, thus the contribution is:

$$\begin{aligned}\mathbf{C}_s = \sum_{t=0}^{T-1} \Bigg(& a^{1,1}_{\mathcal{F}[0]\mathcal{F}[1]} a^{1,2}_{\mathcal{F}[1]\mathcal{F}[2]} \cdots \Delta a^{t+1}_{\mathcal{F}[t]\mathcal{F}[t+1]} a^{0,t+2}_{\mathcal{F}[t+1],\mathcal{F}[t+2]} \cdots a^{0,T}_{\mathcal{F}[T-1],\mathcal{F}[T]} \\ & \mathbf{h}^{1,0}_{\mathcal{F}[0]} \frac{\mathbf{h}^{1,1}_{\mathcal{F}[1]}}{\mathbf{z}^{1,1}_{\mathcal{F}[1]}} \cdots \frac{\mathbf{h}^{1,t}_{\mathcal{F}[t]}}{\mathbf{z}^{1,t}_{\mathcal{F}[t]}} \boldsymbol{\theta}^t \frac{\Delta\mathbf{h}^{t+1}_{\mathcal{F}[t+1]}}{\Delta\mathbf{z}^{t+1}_{\mathcal{F}[t+1]}} \boldsymbol{\theta}^{t+1} \cdots \frac{\Delta\mathbf{h}^{T-1}_{\mathcal{F}[T-1]}}{\Delta\mathbf{z}^{T-1}_{\mathcal{F}[T-1]}} \boldsymbol{\theta}^T \Bigg) / |\mathcal{V}^0 \cup \mathcal{V}^1|\end{aligned} \tag{14}$$

Where, $\mathcal{V}_0$ and $\mathcal{V}_1$ denote the the nodes set of graph G_0 and G_1, respectively.

A.3 Mapping contributions for the graph classification task

In the section 3.2, we show how to calculate the Shapley value, i.e. contribution $\phi_{a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$ of layer edge $a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}$ to $\Delta\mathbf{z}^T_{\mathcal{F}_T}$. Note that the changed layer edge can affect many nodes, not the single node. Thus, in the graph classification task, the contribution matrix of l-th layer edge $a^t_{\mathcal{F}[t-1]\mathcal{F}[t]} \in \Delta\boldsymbol{\mathcal{A}}$ is $\Phi^l \in \mathbb{R}^{|\mathcal{V}^0\cup\mathcal{V}^1|\times c}$, the row vector $\Phi^l_i = \phi_{a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$ denotes the contribution of the l-th layer edge to $\Delta\mathbf{z}^T_{\mathcal{F}_T}$, where the i-th node in the $\mathcal{V}^0 \cup \mathcal{V}^1$ is $\mathcal{F}_T$. Let $\Phi = \sum_{l=1}^{|\Delta\boldsymbol{\mathcal{A}}|} \Phi^l$, the Φ also follows the summation-to-delta property $\sum_{i=1}^{|\mathcal{V}^0\cup\mathcal{V}^1|} \Phi_i = \Delta\mathbf{z} = \mathbf{z}(G_1) - \mathbf{z}(G_0)$

A.4 Selecting the important layer edges

A.4.1 On the link prediction task

For the link prediction, the $\mathbf{z}_{IJ}(G) = [\mathbf{z}_1, \cdots, \mathbf{z}_\ell \cdots, \mathbf{z}_c], \mathsf{Pr}_{IJ}(G) = [\mathsf{Pr}_1(G), \cdots, \mathsf{Pr}_\ell \cdots, \mathsf{Pr}_c(G)]$, Let Φ denotes the contribution matrix of layer edges, where Φ_l represents the contribution of l-th layer edge to $\Delta\mathbf{z}_{IJ}$, and $\Phi_{l,\ell}$ indicates the contribution of l-th layer edge to Δz_ℓ, we can define the following objective function for the link prediction:

$$\begin{aligned}\mathbf{x}^* &= \underset{\substack{\mathbf{x}\in\{0,1\}^{|\Delta\boldsymbol{\mathcal{A}}|} \\ \|\mathbf{x}\|_1 = n}}{\arg\min} \sum_{\ell=1}^{c} \left(-\mathsf{Pr}_\ell(G_1) \sum_{l=1}^{|\Delta\boldsymbol{\mathcal{A}}|} x_l \Phi_{l,\ell} \right) \\ &+ \log \sum_{\ell'=1}^{c} \exp\left(z_{\ell'}(G_0) + \sum_{l=1}^{|\Delta\boldsymbol{\mathcal{A}}|} x_l \Phi_{l,\ell'} \right)\end{aligned} \tag{15}$$

A.4.2 On the graph classification task

For the graph classification, the Φ^l denotes contribution matrix of the l-th layer edge in the $\Delta\boldsymbol{\mathcal{A}}$. The logits of the graph classification $\mathbf{z}_G = [\mathbf{z}_1, \cdots, \mathbf{z}_g \cdots, \mathbf{z}_c]$, the $\mathsf{Pr}(G) = [\mathsf{Pr}_1(G), \cdots, \mathsf{Pr}_g \cdots, \mathsf{Pr}_c(G)]$, because the $\sum_{i=1}^{|\mathcal{V}^0\cup\mathcal{V}^1|} \sum_{l=1}^{|\Delta\boldsymbol{\mathcal{A}}|} \Phi^l_i = \Delta\mathbf{z} = \Delta\mathbf{z}(G_1) - \Delta\mathbf{z}(G_0)$, the objective function for the graph classification task is:

$$\begin{aligned}\mathbf{x}^* &= \underset{\substack{\mathbf{x}\in\{0,1\}^{|\Delta\boldsymbol{\mathcal{A}}|} \\ \|\mathbf{x}\|_1 = n}}{\arg\min} \sum_{g=1}^{c} \left(-\mathsf{Pr}_g(G_1) \sum_{i=1}^{|\mathcal{V}^0\cup\mathcal{V}^1|} \sum_{l=1}^{|\Delta\boldsymbol{\mathcal{A}}|} x_l \Phi^l_{i,g} \right) \\ &+ \log \sum_{g'=1}^{c} \exp\left(z_{g'}(G_0) + \sum_{i=1}^{|\mathcal{V}^0\cup\mathcal{V}^1|} \sum_{l=1}^{|\Delta\boldsymbol{\mathcal{A}}|} x_l \Phi^l_{i,g'} \right)\end{aligned} \tag{16}$$

A.5 Selecting the important layer edges for link prediction

Selecting the important layer edges for link prediction task can be seen in Algorithm 2.

Algorithm 2 Selecting important layer edges to explain evolution of $\mathsf{Pr}(Y|G_0)$ to $\mathsf{Pr}(Y|G_1)$ on the link prediction task

1: **Input**: the source graph G_0 and the destination graph G_1, Pre-trained GNN parameters $\boldsymbol{\theta}$
2: Obtain the layer edges flow set $\Delta\boldsymbol{\mathcal{A}}$
3: Initialize layer edges contribution matrix $\Phi \in \mathbb{R}^{|\Delta\boldsymbol{\mathcal{A}}|\times c}$ as an all-zero matrix
4: Obtain the altered massage flows set $\Delta\boldsymbol{\mathcal{F}}$
5: Given the target edge IJ, $\Delta\boldsymbol{\mathcal{F}} = \{\mathcal{F} : \mathcal{F} \in \Delta\boldsymbol{\mathcal{F}}$ and $(\mathcal{F}[T] = I$ or $\mathcal{F}[T] = J)\}$
6: **for** s for 1 to $|\Delta\boldsymbol{\mathcal{F}}|$ **do**
7: Select the s-th message flow in $|\Delta\boldsymbol{\mathcal{F}}|$ and calculate $\mathbf{C}_s$ according to the Eq. (13)
8: Obtain the changed layer edges set $\Delta\boldsymbol{\mathcal{A}}_{\mathcal{F}}$ on this flow
9: **for** $a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}$ in $\Delta\boldsymbol{\mathcal{A}}_{\mathcal{F}}$ **do**
10: According to the section 3.2 and Eq. (**??**), calculate $\phi_{a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$
11: Let the index of $a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}$ in $\Delta\boldsymbol{\mathcal{A}}$ is l, $\Phi_l = \Phi_l + \phi_{a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$
12: **end for**
13: **end for**
14: Solve Eq. (15) to obtain the important changed layer edges
15: **Output**: The important changed layer edges set

A.5.1 Selecting the important layer edges for graph classification

Selecting the important layer edges for graph classification task can be seen in Algorithm 3.

Algorithm 3 Selecting important layer edges to explain evolution of $\mathsf{Pr}(Y|G_0)$ to $\mathsf{Pr}(Y|G_1)$ on the graph classification tasks

1: **Input**: the source graph G_0 and the destination graph G_1, Pre-trained GNN parameters $\boldsymbol{\theta}$
2: Obtain the layer edges flow set $\Delta\boldsymbol{\mathcal{A}}$
3: Initialize layer edges contribution matrix $\Phi^l \in \mathbb{R}^{|\mathcal{V}^0\cup\mathcal{V}^1|\times c}$ as an all-zero matrix
4: **for** s for 1 to $|\Delta\boldsymbol{\mathcal{F}}|$ **do**
5: Select the s-th message flow in $|\Delta\boldsymbol{\mathcal{F}}|$ and calculate $\mathbf{C}_s$ according to the Eq. (14)
6: obtain the changed layer edges set $\Delta\boldsymbol{\mathcal{A}}_{\mathcal{F}}$ on this flow
7: **for** $a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}$ in $\Delta\boldsymbol{\mathcal{A}}_{\mathcal{F}}$ **do**
8: According to the section 3.2 and Eq. (**??**), calculate $\phi_{a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$
9: Let the index of $a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}$ in $\Delta\boldsymbol{\mathcal{A}}$ is l. Let the index of $\mathcal{F}[T]$ in the $\mathcal{V}^0 \cup \mathcal{V}^1$ is i
10: $\Phi^l_i = \Phi^l_i + \phi_{a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$
11: **end for**
12: **end for**
13: Solving the Eq. (16) to obtain the important changed layer edges
14: **Output**: The important changed layer edges set

A.6 Obtain the important input edges

A.6.1 On the node classification task

Let Φ denotes the contribution matrix of edges, where Φ_l represents the contribution of l-th edge to $\Delta\mathbf{z}_J$, and $\Phi_{l,k}$ indicates the contribution of l-th edge to Δz_k, we can define the following objective function for the node classification:

$$\begin{aligned}\mathbf{x}^* \;=\; & \underset{\substack{\mathbf{x}\in\{0,1\}^{|\Delta\mathcal{E}|}\\ \|\mathbf{x}\|_1=n}}{\arg\min} \;\sum_{k=1}^{c}\left(-\mathsf{Pr}_k(G_1)\sum_{l=1}^{|\Delta\mathcal{E}|} x_l\Phi_{l,k}\right)\\ & + \;\log\sum_{k'=1}^{c}\exp\left(z_{k'}(G_0) + \sum_{l=1}^{|\Delta\mathcal{E}|} x_l\Phi_{l,k'}\right)\end{aligned} \tag{17}$$

A.6.2 On the link prediction task

For the link prediction, the $\mathbf{z}_{IJ}(G) = [\mathbf{z}_1, \cdots, \mathbf{z}_\ell \cdots, \mathbf{z}_c], \mathsf{Pr}_{IJ}(G) = [\mathsf{Pr}_1(G), \cdots, \mathsf{Pr}_\ell \cdots, \mathsf{Pr}_c(G)]$, Let Φ denotes the contribution matrix of edges, where Φ_l represents the contribution of l-th edge to $\Delta\mathbf{z}_{IJ}$, and $\Phi_{l,\ell}$ indicates the contribution of l-th edge to Δz_ℓ, we can define the following objective function for the link prediction:

$$\begin{aligned}\mathbf{x}^* &= \underset{\substack{\mathbf{x}\in\{0,1\}^{|\Delta\mathcal{E}|}\\ \|\mathbf{x}\|_1=n}}{\arg\min} \sum_{\ell=1}^{c}\left(-\mathsf{Pr}_\ell(G_1)\sum_{l=1}^{|\Delta\mathcal{E}|} x_l\Phi_{l,\ell}\right)\\ &+ \log\sum_{\ell'=1}^{c}\exp\left(z_{\ell'}(G_0)+\sum_{l=1}^{|\Delta\mathcal{E}|} x_l\Phi_{l,\ell'}\right)\end{aligned} \tag{18}$$

A.6.3 On the graph classification task

For the graph classification, the Φ^l denotes contribution matrix of the l-th layer edge in the $\Delta\boldsymbol{\mathcal{A}}$. The logits of the graph classification $\mathbf{z}_G = [\mathbf{z}_1, \cdots, \mathbf{z}_g \cdots, \mathbf{z}_c]$, the $\mathsf{Pr}(G) = [\mathsf{Pr}_1(G), \cdots, \mathsf{Pr}_g \cdots, \mathsf{Pr}_c(G)]$, because the $\sum_{i=1}^{|\mathcal{V}^0\cup\mathcal{V}^1|}\sum_{l=1}^{|\Delta\boldsymbol{\mathcal{A}}|}\Phi_i^l = \Delta\mathbf{z} = \Delta\mathbf{z}(G_1) - \Delta\mathbf{z}(G_0)$, the objective function for the graph classification task is:

$$\begin{aligned}\mathbf{x}^* &= \underset{\substack{\mathbf{x}\in\{0,1\}^{|\Delta\boldsymbol{\mathcal{A}}|}\\ \|\mathbf{x}\|_1=n}}{\arg\min} \sum_{g=1}^{c}\left(-\mathsf{Pr}_g(G_1)\sum_{i=1}^{|\mathcal{V}^0\cup\mathcal{V}^1|}\sum_{l=1}^{|\Delta\mathcal{E}|} x_l\Phi_{i,g}^l\right)\\ &+ \log\sum_{g'=1}^{c}\exp\left(z_{g'}(G_0)+\sum_{i=1}^{|\mathcal{V}^0\cup\mathcal{V}^1|}\sum_{l=1}^{|\Delta\mathcal{E}|} x_l\Phi_{i,g'}^l\right)\end{aligned} \tag{19}$$

A.6.4 Selecting the important input edges

Selecting the important input edges for node classification and link prediction can be seen in the Algorithm 4. The selection of important input edges for graph classification can be seen in the Algorithm 5.

A.7 Experiments

A.7.1 Datasets

In the simulated dynamic graphs, we modify edge weights without adding or removing edges. Specifically, given a changed ratio r, we randomly adjust the the weights of $|\mathcal{E}^0| \times r$ edges to create evolving graphs. For the real dynamic graph datasets used in the node classification and link prediction tasks, timestamps allow us to track graph evolution, which includes modifications to edge weights, as well as the addition and deletion of edges. In graph classification, we apply slight perturbations to the graphs You et al. (2018), by randomly adding or removing edges or altering edge weights.

- YelpChi, YelpNYC Rayana & Akoglu (2015): each node represents a review, product, or user. If a user posts a review to a product, there are edges between the user and the review, and between the review and the product. The data sets are used for node classification.
- Pheme Zubiaga et al. (2017) and Weibo Ma et al. (2018): they are collected from Twitter and Weibo. A social event is represented as a trace of information propagation. Each event has a label, rumor or non-rumor. Consider the propagation tree of each event as a graph. The data sets are used for node classification.
- BC-OTC[1] and BC-Alpha[2]: is a who trusts-whom network of bitcoin users trading on the platform. The data sets are used for link prediction.

[1] http://snap.stanford.edu/data/soc-sign-bitcoin-otc.html
[2] http://snap.stanford.edu/data/soc-sign-bitcoin-alpha.html

Algorithm 4 Selecting important input edges to explain evolution of $\Pr(Y|G_0)$ to $\Pr(Y|G_1)$ on the node classification and link prediction tasks

Input: the source graph G_0 and the destination graph G_1, Pre-trained GNN parameters $\boldsymbol{\theta}$
Obtain the changed edges set $\Delta\mathcal{E} = \{a_{UV} : a^0_{UV} \neq a^1_{UV}, t \in \{1, \ldots, T\}, U, V \in \mathcal{V}^0 \cup \mathcal{V}^1\}$
Initialize layer edges contribution matrix $\Phi \in \mathbb{R}^{|\Delta\mathcal{E}| \times c}$ as an all-zero matrix
Obtain the altered massage flows set $\Delta\boldsymbol{\mathcal{F}} = \{\mathcal{F} : \mathcal{F} = (\mathcal{F}[0], \ldots, \mathcal{F}[t] \ldots \mathcal{F}[T]), a^{0,t}_{\mathcal{F}[t-1]\mathcal{F}[t]} \neq a^{1,t}_{\mathcal{F}[t-1]\mathcal{F}[t]}, t = 1, \ldots, T\}$
if The node classification task **then**
    Given the target node J, $\Delta\boldsymbol{\mathcal{F}} = \{\mathcal{F} : \mathcal{F} \in \Delta\boldsymbol{\mathcal{F}}$ and $\mathcal{F}[T] = J\}$
else if The link prediction task **then**
    Given the target edge IJ, $\Delta\boldsymbol{\mathcal{F}} = \{\mathcal{F} : \mathcal{F} \in \Delta\boldsymbol{\mathcal{F}}$ and $(\mathcal{F}[T] = I$ or $\mathcal{F}[T] = J)\}$
end if
for $\mathcal{F}$ in $|\Delta\boldsymbol{\mathcal{F}}|$ **do**
    According to the Eq. (7) (node classification) or Eq. (13) (link prediction), calculate the message flow contribution $\mathbf{c}$
    obtain the changed edges set $\Delta\mathcal{E}_{\mathcal{F}} = \{a_{\mathcal{F}[t-1]\mathcal{F}[t]} : a^0_{\mathcal{F}[t-1]\mathcal{F}[t]} \neq a^1_{\mathcal{F}[t-1]\mathcal{F}[t]}\}$ on this flow
    for $a_{\mathcal{F}[t-1]\mathcal{F}[t]}$ in $\Delta\mathcal{E}_{\mathcal{F}}$ **do**
        According to the Section 3.2 and Eq. (**??**), calculate $\phi_{a_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$.
        Let the $a_{\mathcal{F}[t-1]\mathcal{F}[t]}$ is the l-th edge in $\Delta\mathcal{E}$, $\Phi_l = \Phi_l + \phi_{a_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$
    end for
end for
Solving the Eq. (17) (node classification) or Eq. (18) (link prediction) to obtain the important changed input edges
Output: The important changed input edges set

Algorithm 5 Selecting the important input edges to explain evolution of $\Pr(Y|G_0)$ to $\Pr(Y|G_1)$ on the graph classification tasks

Input: the source graph G_0 and the destination graph G_1, Pre-trained GNN parameters $\boldsymbol{\theta}$
Obtain the layer edges flow set $\Delta\boldsymbol{\mathcal{A}} = \{a^t_{UV} : a^{0,t}_{UV} \neq a^{1,t}_{UV}, t \in \{1, \ldots, T\}, U, V \in \mathcal{V}^0 \cup \mathcal{V}^1\}$
Obtain the altered massage flows set $\Delta\boldsymbol{\mathcal{F}} = \{\mathcal{F} : \mathcal{F} = (\mathcal{F}[0], \ldots, \mathcal{F}[t] \ldots \mathcal{F}[T]), a^{0,t}_{\mathcal{F}[t-1]\mathcal{F}[t]} \neq a^{1,t}_{\mathcal{F}[t-1]\mathcal{F}[t]}, t = 1, \ldots, T\}$
for l for 1 to $|\Delta\boldsymbol{\mathcal{A}}|$ **do**
    Initialize layer edges contribution matrix $\Phi^l \in \mathbb{R}^{|\mathcal{V}^0 \cup \mathcal{V}^1| \times c}$ as an all-zero matrix
end for
for $\mathcal{F}$ in $|\Delta\boldsymbol{\mathcal{F}}|$ **do**
    According to the Eq. (14), calculate the message flow contribution $\mathbf{c}$
    obtain the changed edges set $\Delta\mathcal{E}_{\mathcal{F}} = \{a_{\mathcal{F}[t-1]\mathcal{F}[t]} : a^0_{\mathcal{F}[t-1]\mathcal{F}[t]} \neq a^1_{\mathcal{F}[t-1]\mathcal{F}[t]}\}$ on this flow
    for $a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}$ in $\Delta\mathcal{E}_{\mathcal{F}}$ **do**
        According to the section 3.2 and Eq. (**??**), calculate $\phi_{a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$
        Let the $a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}$ is the l-th layer edge in $\Delta\mathcal{E}$, $\mathcal{F}[T]$ is the i-th node in the $\mathcal{V}^0 \cup \mathcal{V}^1$, $\Phi^l_i = \Phi^l_i + \phi_{a^t_{\mathcal{F}[t-1]\mathcal{F}[t]}}(\mathcal{F})$
    end for
end for
Solving the Eq. (19) to obtain the important changed input edges
Output: The important changed input edges set

- UCI[3]: is an online community of students from the University of California, Irvine, where in the links of this social network indicate sent messages between users. The data sets are used for link prediction.
- MUTAG Debnath et al. (1991): A molecule is represented as a graph of atoms where an edge represents two bounding atoms.
- ClinTox Gayvert et al. (2016):compares drugs approved through FDA and drugs eliminated due to the toxicity during clinical trials.
- IMDB-BINARY is movie collaboration datasets. Each graph corresponds to an ego-network for each actor/actress, where nodes correspond to actors/actresses and an edge is drawn betwen two actors/actresses if they appear in the same movie.Each graph is derived from a pre-specified genre of movies, and the task is to classify the genre graph it is derived from.
- REDDIT-BINARY is balanced datasets whereeach graph corresponds to an online discussion thread and nodes correspond to users. An edge was drawn between two nodes if at least one of them responded to another's comment. The task is to classify each graph to a community or a subreddit it belongs to.

Table 2: The details of datasets

Datasets	Nodes(Avg. Nodes)	Edges(Avg. Edges)	task
YelpChi	105,659	375,239	node classification
YelpNYC	520,200	1,956,408	node classification
weibo	4,657	-	node classification
pheme	5,748	-	node classification
BC-OTC	5,881	35,588	link prediction
BC-Alpha	3,777	24,173	link prediction
UCI	1,899	59,835	link prediction
MUTAG	17.93	19.79	graph classification
ClinTox	26.1	55.5	graph classification
IMDB-BINARY	19.8	193.1	graph classification
REDDIT-BINARY	429.6	995.5	graph classification

Table 3: The changed ration r on different datasets

YelpChi	YelpNYC	Weibo	Pheme	BC-OTC	BC-Alpha	UCI	MUTAG	ClinTox	IMDB-BINARY	REDDIT-BINARY
1	1	1	1	0.5	0.6	0.4	1	1	1	1

A.7.2 Basslines

- **GNNExplainer** is designed to explain GNN predictions for node and graph classification on static graphs. We train the explainer on graphs G_0 and G_1 to obtain the edges contribution Φ^0 and Φ^1. The final edges contribution is given by $\Phi = \Phi^1 - \Phi^0$ if the predicted class on G_0 and G_1 are different. Otherwise, $\Phi = \Phi^1$. The top-K edges are selected based on Φ as the explanations.
- **PGExplainer** learns approximated discrete masks for edges to explain the predictions, with important edges selected in the same manner as GNNExplainer.
- **GNN-LRP** utilizes the back-propagation attribution method LRP to GNN Schnake et al. (2020), attributing the class probability $\Pr(Y = k|G_1)$ to input neurons regardless of $\Pr(Y|G_0)$, thereby obtaining contribution scores for message flows. It uses a summation function to map these contributions to edges, with edge selection consistent with GNNExplainer.
- **DeepLIFT** Shrikumar et al. (2017) attributes the log-odd between two probabilities $\Pr(Y = k|G_0)$ and $\Pr(Y = k'|G_1)$, where $k \neq k'$, to the message flows. Then it uses a summation function to obtain contributions of edges. The edge selection process is consistent with GNNExplainer.

[3]http://konect.cc/networks/opsahl-ucsocial

- **FlowX** applies the Shapley value to derive initial contributions of message flows, subsequently training these scores by defining loss functions. A summation function is employed to map contributions to edges, with edge selection aligned with GNNExplainer.

A.7.3 EXPERIMENTAL SETUP

We trained the two layers GNN. utilizing element-wise sum as the aggregation function f_{AGG}. The logit for node J is denoted by $z_J(G)$. For node classification, $z_J(G)$ is mapped to the class distribution through the softmax function. For the link prediction, we concatenate $z_I(G)$ and $z_J(G)$ as the input to a linear layer to obtain the logits, which are then mapped to the probability of the existence of the edge (I, J). For the graph classification task, the average pooling of $z_J(G)$ across all nodes in G can produce a single vector representation $z(G)$ for classification. It can be mapped to the class probability distribution through the softmax function. During training, we set the learning rate to 0.01, the dropout rate to 0.2 and the hidden size to 16. The model is trained and then fixed during the prediction and explanation stages.

A.7.4 THE PREDEFINED SPARSITY

On the real dynamic graphs, the sparsity of explanations across various datasets and tasks is illustrated in Table 4. The sparsity of simulated dynamic graphs is illustrated in Table 5. The sparsity is small, but our method can also achieve the better performance than the baselines.

Table 4: The sparsity of explanations on real dynamic graph datasets

Datasets	Sparsity level 1	Sparsity level 2	Sparsity level 3	Sparsity level 4	Sparsity level 5
YelpChi	0.996	0.992	0.988	0.994	0.98
YelpNYC	0.998	0.997	0.996	0.995	0.994
weibo	0.996	0.993	0.99	0.986	0.982
pheme	0.98	0.96	0.94	0.92	0.9
BC-OTC	0.996	0.995	0.994	0.993	0.992
BC-Alpha	0.995	0.994	0.993	0.992	0.991
UCI	0.998	0.997	0.996	0.994	0.992
MUTAG	0.988	0.976	0.964	0.952	0.94
ClinTox	0.991	0.982	0.973	0.964	0.954
IMDB-BINARY	0.996	0.991	0.988	0.984	0.98
REDDIT-BINARY	0.998	0.997	0.996	0.995	0.994

Table 5: The sparsity of explanations on different simulated graph datasets

Datasets	Sparsity level 1	Sparsity level 2	Sparsity level 3	Sparsity level 4	Sparsity level 5
YelpChi	0.999	0.998	0.997	0.996	0.995
YelpNYC	0.9994	0.9988	0.9981	0.9975	0.9965
weibo	0.9972	0.9945	0.992	0.989	0.986
pheme	0.982	0.963	0.945	0.927	0.908
BC-OTC	0.967	0.95	0.935	0.918	0.9
BC-Alpha	0.95	0.91	0.87	0.83	0.79
UCI	0.999	0.998	0.997	0.996	0.995
MUTAG	0.988	0.976	0.964	0.952	0.94
ClinTox	0.99	0.98	0.97	0.96	0.95
IMDB-BINARY	0.996	0.992	0.988	0.984	0.98
REDDIT-BINARY	0.998	0.996	0.994	0.992	0.99

A.7.5 PERFORMANCE EVALUATION AND COMPARISON

We compare the performance of the methods across three tasks: node classification, link prediction and graph classification in simulate dynamic graph scene, as illustrated in Figure 7. For each dataset, we report the average KL over target nodes/edges/graphs. From Figure 7, we can see that our method AxiomLayeredge has the smallest KL across all levels of explanation sparsity and datasets and tasks, with exception of Weibo, Pheme and certain sparsity levels of YelpNYC dataset.

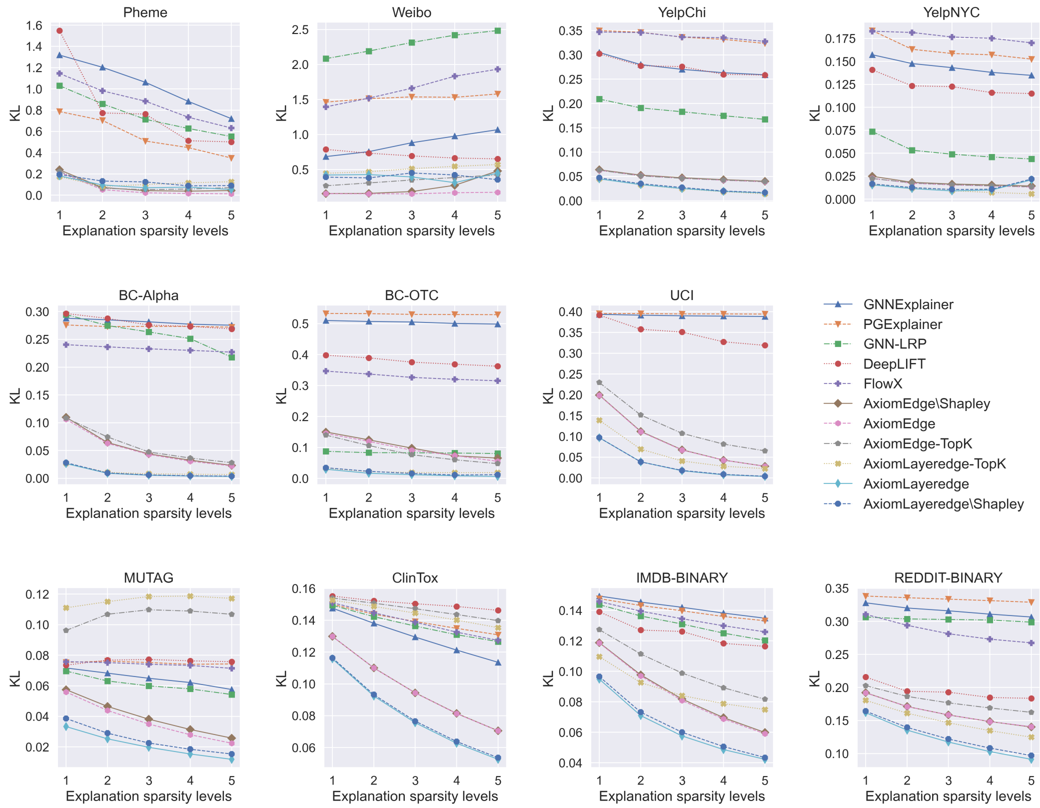

Figure 7: Performance in KL as $G_0 \to G_1$. Each column corresponds to a different dataset. The first, second and third rows represent node classification, link prediction and graph classification tasks, respectively.

In datasets with dense graph structures (YelpChi, YelpNYC, BC-Alpha, BC-OTC, UCI, IMDB-BINARYdand REDDIT-BINAYR), the AxiomLayeredge-TopK method ranks third. This indicates that our designed message flow contribution value Algorithm can effectively explain the dynamic graphs. In seven experimental settings (Weibo, YelpChi, YelpNYC, BC-Alpha, UCI, MUTAG, ClinTox), our method AxiomLayeredge along with its variants AxiomEdge, AxiomEdge\Shapley, AxiomLayeredge\Shapley outperform the GNNLRP, DeepLIFT, GNNExplainer, PGExplainer and FlowX methods. This demonstrates that our proposed methods more effectively explain the evolution of $\Pr(Y|G_0;\boldsymbol{\theta})$ to $\Pr(Y|G_1;\boldsymbol{\theta})$, while methods designed for static graph struggle to identify salient edges that explain changes in the predicted probability distribution.