ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Enriching Graph Topology Representations with Line Graph Transformations

Paolo Frazetto[1], Luca Pasa[1], Nicolò Navarin[1], Alessandro Sperduti[1,2,3] *

1- University of Padua, Padua, Italy
2- Fondazione Bruno Kessler (FBK), Trento, Italy
3- University of Trento, Trento, Italy

Abstract. Many Graph Neural Networks (GNNs) in the literature are based on message-passing, which introduces a strong learning bias that may fail to capture critical relational information encoded in the edges of the graph, particularly in tasks where the structural role of edges is as significant as that of nodes, such as in chemical molecular analysis or social network dynamics. We propose a novel architecture inspired by line graph theory that explicitly models edge adjacencies, iteratively transforming a graph into its corresponding *line graph*. Differently from message-passing, the iterative application of this transformation enables the exchange of information among non-adjacent nodes, allowing for the capture of complex topological dependencies, which standard GNNs overlook. Experiments on standard benchmarks show promising results.

1 Introduction

Despite their effectiveness, the vast majority of existing graph neural models, e.g., Graph Convolutional Networks (GCNs) [1], operate on local information aggregation, where each node aggregates (possibly weighted [2]) information only from its immediate neighbors. This approach may limit a model's ability to capture long-range dependencies effectively and create some undesirable phenomena, such as over-smoothing.

Line graphs are intended to enhance connectivity within the graph and facilitate long-range interactions; they can provide an alternative and intriguing mechanism to deal with a graph's topological information [3]. In this paper, we propose a novel framework based on the idea of iteratively computing line graphs, mapping an input graph to a sequence of line graphs that allow the exchange of information across both intra-graph and inter-graph levels. This approach aims to learn rich hierarchical embeddings capable of modeling complex topological dependencies. We empirically evaluate the proposed approach on several commonly adopted graph classification benchmarks. We conducted experiments on several standard benchmark datasets

*This research was supported by: the Italian Ministry of University and Research – NextGenerationEU, Mission 4 Component 2 Investment 1.3 - Call for tender No. 341 of March 15, 2022, project code PE0000013, Concession Decree No. 1555 of October 11, 2022, CUP C63C22000770006, project title "Future AI Research (FAIR) - Spoke 2 Integrative AI - Symbolic conditioning of Graph Generative Models (SymboliG)"; the Italian Ministry of University and Research under the PRIN program – Progetti di Rilevante Interesse Nazionale – PRIN 2022 (Secretary-General's Decree n. 1401 of 18/09/2024) – CUP B53C24006570006, project title "DEEP-GRAPH: Design and Theory of Deep Graph Learning"; Deep Learning in Structured Domains for Functional Neuroimaging Data Analysis - jointly funded by the University of Lausanne and the University of Padua, CUP C93C24004750005.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

aimed at identifying the viability of this research direction. The results are promising and show that the proposed method performs comparably, and in some cases better, compared to commonly adopted graph neural networks based on message-passing, such as GCN and its variants.

2 Background and Related Works

Graph Neural Networks. GNNs have emerged as the preferred machine learning model for addressing graph-related problems [4, 5]. The central idea is to design a neural architecture that aligns with the graph's topology. Let $G=(V,E,\mathbf{X})$ be a graph, where $V=\{v_1,...,v_n\}$ denotes the set of nodes, $E\subseteq V\times V$ is the set of edges, and $\mathbf{X}\in\mathbb{R}^{n\times s}$ is a multivariate signal on the graph nodes. We define $\mathbf{A}\in\mathbb{R}^{n\times n}$ as the adjacency matrix of the graph, where $a_{ij}=1\iff(v_i,v_j)\in E$. With $\mathcal{N}(v)$, we denote the set of nodes adjacent to node v. A GNN leverages $\mathbf{X}$ and A to learn a representation $\mathbf{h}_v\in\mathbb{R}^m$, $\forall\, v\in V$. In general, the computation of $\mathbf{h}_v$ is typically structured into two key steps, namely *aggregation* $\mathcal{A}$ and *combination* $\mathcal{C}$: $\mathbf{h}_v=\mathcal{C}(\mathbf{x}_v,\mathcal{A}(\mathbf{x}_u:u\in\mathcal{N}(v)\}))$. $\mathcal{A}$ collects information from the neighboring nodes $\mathcal{N}(v)$, while $\mathcal{C}$ integrates this information with the node's own features $\mathbf{x}_v$. The choice of $\mathcal{A}$ and $\mathcal{C}$ determines the specific type of graph convolution. This mechanism is referred to as *message passing* [6].

Line Graph. Given a graph G with at least one edge, its line graph, denoted by $L(G)$, is a graph whose vertices are the edges of G, where two of these vertices are adjacent if the corresponding edges are incident in G [3]. Since the line graph of a graph is itself a graph, this process can be repeated, generating a sequence of line graphs. Formally, the iterated line graphs of $G^{(0)}$ are defined as $G^{(1)}=L(G^{(0)}),...,G^{(n)}=L(L_{n-1}(G^{(0)}))$. If G is a non-trivial connected graph, its line graph $L(G)$ will also be connected. A key property is that the transformation into a line graph tends to redistribute the degrees of the nodes [3], potentially leading to a graph with stronger local connectivity and a more balanced degree distribution compared to the original graph. In contrast, in message-passing models the spread of information through the graph is determined solely by the adjacency matrix. In a line graph, the *incidence matrix* $\mathbf{B}(G)\in\mathbb{R}^{n\times m}$ has elements $b_{ij}=1\iff v_i$ that are linked to the edge e_j,0 otherwise. This matrix can be used to define the adjacency matrix of the line graph, as $\mathbf{A}(G)=\mathbf{B}(G)\mathbf{B}(G)^T-\mathbf{D}$, and $\mathbf{A}(L(G))=\mathbf{B}(G)^T\mathbf{B}(G)-2\mathbf{I}$, where $\mathbf{D}\in\mathbb{R}^{n\times n}$ is the diagonal degree matrix of G, and $\mathbf{I}$ is the identity matrix [3].

Line Graph-based Neural Networks. Although line graphs have the potential to enhance graph-based learning, their application remains largely unexplored. Cai *et al.* [7] introduce LGNN (Line Graph Neural Network), a framework designed for link prediction. In [8], the authors present a GNN model based on a family of multi-scale graph operators that utilizes line graphs for the community detection problem. Another approach to utilizing line graphs in defining a novel GNN model is presented in [9], where the authors introduce Line Graph Contrastive Learning (LineGCL) for node classification tasks. Liang *et al.* [10] recently introduced Line Graph Neural

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Networks for Link Weight Prediction (LGLWP), a method that leverages line graph transformation and GCNs to learn link features.

Although these methods are formulated in terms of line graphs, they nonetheless depend on message-passing as they apply it directly on the line graphs representations. In contrast, our method seeks to construct a mechanism that acts as an alternative to message-passing, thereby inducing a distinct implicit learning bias.

3 Iterative Line Graph Neural Network

In this section, we present the *Iterative Line Graph Neural Network* (ILGNN), a model designed to capture higher-order structural information in graphs by propagating features through a sequence of line graphs. In contrast to conventional GNNs that rely on message passing over a fixed graph topology, ILGNN repeatedly transforms node features into edge features, which then serve as node representations in the subsequent line graph iteration.

Given an input graph $G^{(0)}$, we construct a sequence of L line graphs $\{G^{(1)},...,G^{(L)}\}$. For any level k, the graph $G^{(k+1)}$ is the line graph of $G^{(k)}$, where the node set $V^{(k+1)}$ corresponds to the edge set $E^{(k)}$, and two nodes in $G^{(k+1)}$ are connected if their corresponding edges in $G^{(k)}$ share a common endpoint. This preprocessing step is performed prior to training to ensure efficiency.

ILGNN Architecture. To propagate the features across the iterated line graph transformations, the input features $\mathbf{X}$ of the root graph are first projected into a hidden dimension d by means of a linear layer made of Layer Normalization, GELU activation function, and dropout: $\mathbf{H}^{(0)} = \text{Dropout}(\text{GELU}(\text{LayerNorm}(XW_0 + b_0)))$,where $W_0 \in \mathbb{R}^{F_{in} \times d}$ and $b_0 \in \mathbb{R}^d$ are learnable parameters. For each subsequent level $k \in \{1,...,L\}$, the node features of $G^{(k+1)}$ are derived from the node features of $G^{(k)}$. Since a node $w \in V^{(k+1)}$ corresponds to an edge $e_{uv} = (u,v) \in E^{(k)}$, its representation $h_w^{(k+1)}$ is derived by aggregating the features of its constituent nodes u and v from the previous level. This process effectively generates a novel representation for the edge, which is in turn treated as a node in the subsequent line graph. The model processes the hierarchy sequentially, generating node embeddings $\mathbf{H}^{(k)}$ for each graph level $G^{(k)}$. Specifically, let $h_u^{(k)}$ be the feature vector of node u in graph G_k. For every edge $e_{uv} = (u,v)$ in G_k, the model first retrieves the embeddings of the two connected nodes, $h_u^{(k)}$ and $h_v^{(k)}$. Then, to ensure permutation invariance with respect to the ordering of nodes (i.e., (u,v) is the same as (v,u)), the model constructs two concatenated vectors: $z_{uv} = [h_u^{(k)} \,\|\, h_v^{(k)}], \quad z_{vu} = [h_v^{(k)} \,\|\, h_u^{(k)}]$which are processed by a Feed Forward Neural Network (FFN) that acts as the learnable function deciding how to combine endpoint features into an edge feature. The outputs are averaged to produce the final feature vector $h_{e_{uv}}^{(k+1)}$ for the corresponding node in the next level, computing $h_{e_{uv}}^{(k+1)} = \frac{1}{2}(\text{FFN}(z_{uv}) + \text{FFN}(z_{vu}))$

A global pooling operation aggregates information to form a graph-level representation. The model has been tested with two pooling strategies: GraphNorm [11] followed by a concatenation of *mean*, *max*, and *sum* pooling for each level; or a *Set*

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Transformer [12] has been used to aggregate features across all nodes, treating the entire hierarchy simultaneously as a set $\{H^{(0)},...,H^{(L)}\}$. The final classification is performed by a FFN for the given task.

4 Experimental Evaluation

We compared our approach against four representative graph convolution operators: GCN [1], GraphConv [13], GAT [2], and GNN-FiLM [14]. To ensure a fair comparison, we performed rigorous fine-tuning and validation of hyperparameters for all models, adhering to established guidelines for GNN architecture design [15].

Training was conducted using the ADAM optimizer with a cosine learning rate schedule (starting at 0.01 and annealed to 0, without restarting), along with an L_2 weight decay of 5×10^{-4}. The batch size was set to 32 for all datasets, and each experiment was run for up to 400 epochs.

Each baseline tested architecture incorporated Multilayer Perceptron layers before and after the GC operator layers [15]. We adopted an 80%/10%/10% train/validation/test split, ensuring that every configuration was run five times. The final evaluation metrics were computed as the average performance on the test set at the best validation epoch. The code and the hyperparameter grid used for the grid-search are publicly available[1].

Datasets. All the methods considered in this study were empirically validated on five widely used graph classification benchmark datasets. Specifically, we utilized three datasets focused on bioinformatics problems: NCI1 [16], PROTEINS [17], D&D [18]. Additionally, we used two large social network datasets: IMDB-B and IMDB-M [19].

Results. Table 1 reports the average accuracy achieved by the proposed ILGNN and the baselines. Our approach, based on line graphs, achieves superior performance in two out of the five datasets, namely DD and PROTEINS, and it is the second best in IMDB-M. Conversely, the GraphConv model, which leverages message-passing, obtains better results on the remaining three datasets, while the other baseline architectures are inferior. This dichotomy suggests that the iterated line graph provides a unique perspective for learning hidden representations, offering an alternative to the widely adopted message-passing GNNs. Additionally, the final aggregation with Set Transformer is validated as the best configuration only for the IMDB-M dataset, while standard graph pooling yields the best outcomes for the other cases, demonstrating that the line graph representations are expressive enough without the additional transformation provided by the transformer.

5 Conclusion

In this paper, we propose a novel Graph Neural Network that introduces a new methodology for computing topological node embeddings, providing an effective

[1] `https://github.com/paolofraz/ILGNN`

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Table 1: Accuracy and standard deviation, in percentages, on the test set for the best-validated models on all the datasets. The best performances are highlighted in bold and the second-best are underlined.

Dataset \ GNN	GNN-FiLM	GAT	GCN	GraphConv	ILGNN
DD	77,9±2,1	77,6±2,0	77,6±3,4	77,6±2,4	**78,4±3,6**
IMDB-B	53,7±4,8	55,7±2,5	71,7±5,3	**73,0±3,3**	71,7±6,1
IMDB-M	41,1±7,5	40,0±2,2	50,5±2,7	**50,9±2,5**	49,2±2,9
NCI1	79,0±2,1	80,2±2,0	80,6±1,4	**81,7±1,7**	77,7±1,3
PROTEINS	73,8±3,0	74,1±2,6	73,8±2,3	75,0±4,1	**77,2±2,9**

alternative to the traditional message-passing-based GNN embeddings. The proposed ILGNN model is grounded in line graph theory, and unlike other line-graph-based approaches in the literature, ILGNN eliminates the need for message passing and instead leverages line graph representations to compute topological node aggregation, creating a novel approach with a different learning bias. We evaluated ILGNN on five standard graph classification benchmarks and compared it with commonly adopted message-passing-based graph neural networks. The results demonstrate the promise of the proposed approach, showing comparable performance and, in specific cases, outperforming the considered baselines.

Future work will explore the theoretical properties of iterated line graphs for the expressivity of GNNs, and their application to edge-centric tasks such as link prediction.

References

[1] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*, pages 1–14, 2017.

[2] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[3] Lowell W Beineke and Jay S Bagga. *Line graphs and line digraphs*. Springer, 2021.

[4] Alessandro Sperduti and Antonina Starita. Supervised neural networks for the classification of structures. *IEEE Trans. Neural Networks*, 8(3):714–735, 1997.

[5] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[6] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1263—-1272, apr 2017.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

[7] Lei Cai, Jundong Li, Jie Wang, and Shuiwang Ji. Line graph neural networks for link prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5103–5113, 2021.

[8] Zhengdao Chen, Lisha Li, and Joan Bruna. Supervised community detection with line graph neural networks. In *ICLR*, 2019.

[9] Mingyuan Li, Lei Meng, Zhonglin Ye, Yuzhi Xiao, Shujuan Cao, and Haixing Zhao. Line graph contrastive learning for node classification. *Journal of King Saud University-Computer and Information Sciences*, 36(4):102011, 2024.

[10] Jinbi Liang, Cunlai Pu, Xiangbo Shu, Yongxiang Xia, and Chengyi Xia. Line graph neural networks for link weight prediction. *Physica A: Statistical Mechanics and its Applications*, page 130406, 2025.

[11] Tianle Cai, Shengjie Luo, Keyulu Xu, Di He, Tie-yan Liu, and Liwei Wang. Graphnorm: A principled approach to accelerating graph neural network training. In *International Conference on Machine Learning*. PMLR, 2021.

[12] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pages 3744–3753. PMLR, 2019.

[13] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4602–4609, oct 2019.

[14] Marc Brockschmidt. Gnn-film: Graph neural networks with feature-wise linear modulation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1144–1152. PMLR, 2020.

[15] Jiaxuan You, Rex Ying, and Jure Leskovec. Design space for graph neural networks. In *NeurIPS*, 2020.

[16] Nikil Wale, Ian A Watson, and George Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowledge and Information Systems*, 14(3):347–375, 2008.

[17] Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1):i47–i56, 2005.

[18] Paul D Dobson and Andrew J Doig. Distinguishing enzyme structures from non-enzymes without alignments. *Journal of molecular biology*, 330(4):771–783, 2003.

[19] Pinar Yanardag and SVN Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1365–1374, 2015.