# Topology-Aware Node Dropping Augmentation for Graph Classification

Waqar Ali
*Ca' Foscari University of Venice*
*Venice, Italy*
*ZHAW School of Engineering*
*Winterthur, Switzerland*
aliw@zhaw.ch

Sebastiano Vascon
*Ca' Foscari University of Venice*
*ECLT European Centre for Living Technology*
*Venice, Italy*
sebastiano.vascon@unive.it

Thilo Stadelmann
*ZHAW School of Engineering*
*Winterthur, Switzerland*
*ECLT European Centre for Living Technology*
*Venice, Italy*
stdm@zhaw.ch

Marcello Pelillo*
*College of Mathematical Medicine*
*Zhejiang Normal University*
*Jinhua, China*
*Ca' Foscari University of Venice*
*ECLT European Centre for Living Technology*
*Venice, Italy*
pelillo@unive.it
marcello.pelillo@zjnu.edu.cn

***Abstract*—Graph augmentations effectively enhance the robustness and generalization of Graph Neural Networks (GNNs), particularly for graph classification tasks. However, existing augmentation methods, like NodeDrop, randomly drop a certain portion of nodes to generate augmented graphs without preserving the essential topological structures of the original graph, potentially modifying label information. To address this issue, we introduce a novel Node-Dropping Augmentation (NDAUG) method for graph classification tasks. Our method leverages node degree as a criterion to selectively drop less important nodes (low-degree) and preserve essential graph structures, generating diverse and informative augmented graphs. Further, in the case of isolated nodes, we develop a structure learning method to reconnect these isolated nodes by learning attention-based relationships between nodes. Experiments demonstrate that combining the proposed NDAUG with existing GNN models yields an average improvement of 2–5% accuracy on eight graph classification benchmarks compared to the state-of-the-art baselines.**

***Index Terms*—Node Augmentation, Graph Neural Network, Graph Classification.**

## I. Introduction

Recently, Graph Neural Networks (GNNs), a specialized form of deep learning designed for graph-structured data, have shown effectiveness in various tasks of classifying graphs and learning graph representations, including predicting chemical molecular properties and analyzing social networks [1], [2]. However, similar to deep learning models in image processing, GNNs are prone to overfitting, particularly when dealing with limited datasets [3], [4]. Data augmentation methods are known for their efficiency and effectiveness in mitigating overfitting issues in deep learning networks [5], [6]. These methods generate new synthetic samples from the existing training data, providing a straightforward and cost-efficient method to enhance the generalization of a deep model. Data augmentation has proven helpful in the computer vision domain [3], [7], but applying these techniques to graph-structured data presents unique challenges due to the graphs' irregular structures [8]. Some recent works focused on developing node feature-based graph mixup augmentation methods for node-level tasks [9], others like Kong et al. [10] recommend enhancing node features through adversarial learning. While graph structural learning augmentation methods [11], [12], [5] usually modify the graph structure by randomly dropping/adding nodes or edges and generating new augmented graphs for graph-level tasks. However, current graph learning augmentation methods, such as random node-dropping, often fail to preserve the original graph's essential topological structures during the augmentation process, potentially affecting the performance of the graph classification tasks. This random node-dropping method can also disconnect closely related nodes, generating isolated nodes in the augmented graph [11], [13], which may affect the efficiency of the GNN's message-passing mechanism.

To this end, we introduce a novel node-degree based Node-Dropping Augmentation (NDAUG) method for graph classification tasks to address the issues in existing augmentation approaches. The node degree is an important concept in graph theory because nodes with higher degrees often correspond to critical points in various graphs, such as road, social, or protein-protein interaction graphs, which are essential to identifying the structure and functionality of the entire graph [14]. Our method generates augmented graphs by leveraging the node-degree concept to remove less important low-degree nodes while preserving significant topological struc-

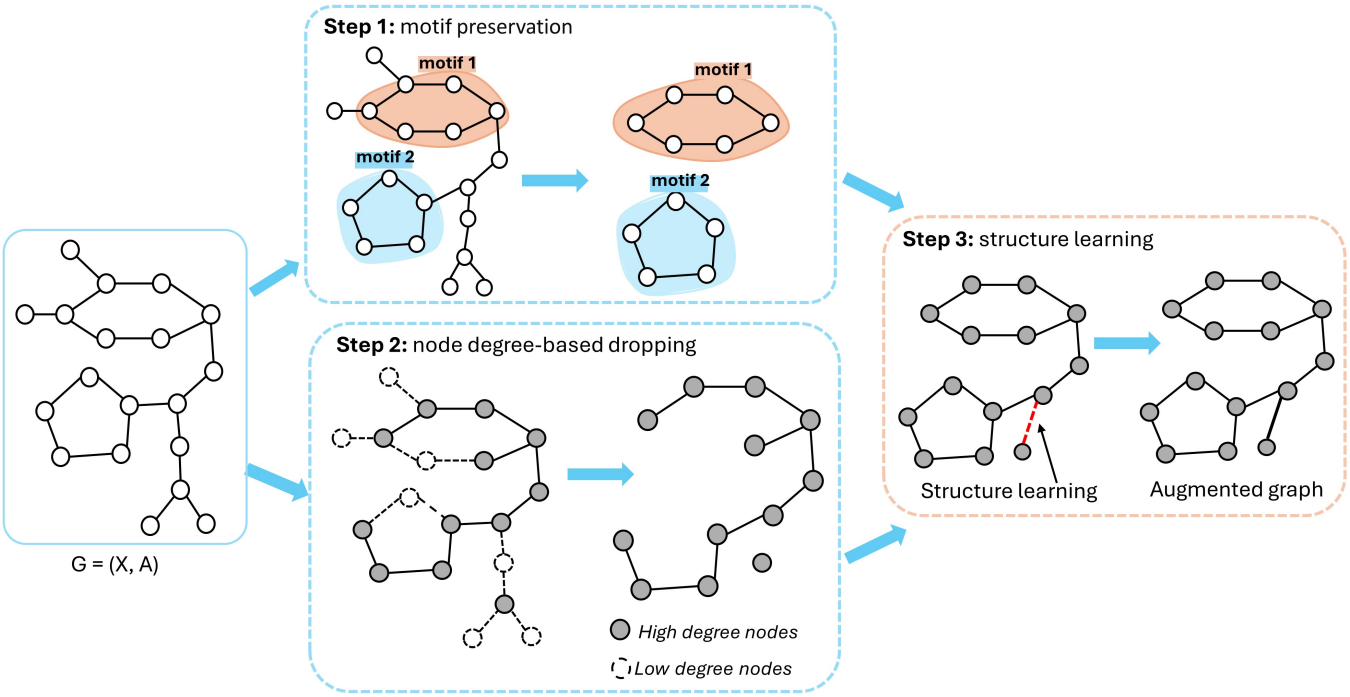



Fig. 1: The pipeline of the proposed NDAUG method. The initial step identifies the important structural motifs of the input graph. Step 2 removes the low-degree nodes while maintaining the key topological structures formed by high-degree nodes. The last step generates the final augmented graph to preserve the identified significant motif structures of step 1 and applies a structure learning method to retain the connectivity of the augmented graph by reconnecting any isolated nodes resulting from the node-dropping process.

tures formed by high-degree nodes, thereby maintaining the essential characteristics of the original graph. However, in applications such as molecular datasets with toxic/non-toxic compounds, the functional groups often contain low-degree nodes such as benzene or carbon motif ring structures [12]. To handle such cases, NDAUG identifies and preserves these significant structures, ensuring the retention of vital low-degree nodes. Additionally, we propose an attention mechanism to reconnect isolated nodes that may result from the node-dropping process, thus maintaining the connectivity in the graph and enhancing the performance of GNN layers. To summarize, the main contributions of this paper are as follows:

- We propose a novel NDAUG augmentation method that uses node degree to remove less important nodes while preserving key topological structures.
- We introduce an attention-based structure learning method to reconnect isolated nodes, maintaining graph connectivity within the augmented graphs and enhancing GNN performance.
- Experiments on eight benchmark datasets show that NDAUG outperforms existing augmentation methods with a $2-5\%$ improvement.

## II. Related Work

Recently, GNNs [1], [15] have emerged as a powerful tool and attained significant achievements in graph classification tasks. Despite GNNs' success, data availability is a significant limitation in many graph classification problems [12], [16]. For instance, GNNs have been widely used in predicting molecular properties, where obtaining labeled molecule data often involves complex manual laboratory procedures. This leads to a lack of adequately labeled samples for GNNs to attain a promising prediction performance.

Data augmentation methods are noted for their efficiency and effectiveness in generating new synthetic samples from existing training data, thereby enhancing the generalization capabilities of deep models [17], [18]. This strategy is preferred over more resource-intensive methods like gathering extra real data or making significant changes to the model architecture or training algorithms. Data augmentation methods have proven helpful in the fields of computer vision (CV) and neural language processing [3]. However, applying such techniques to graphs is more complex due to their non-Euclidean nature, where nodes are irregularly connected by edges [19], [20], presenting unique challenges in augmentation. Recent works have focused on developing node feature space-based augmentation methods for node-level tasks [9], [10], and a limited number

of attempts [21] have been undertaken for graph classification tasks. For example, DropEdge [5] employs a random method to remove a uniform portion of edges and generate augmented graphs to enhance the robustness of the GNN model during test-time inference, M-evolve [22] methodology uses motif-similarity mapping methods to add or remove edges connecting nodes that are predicted to have similar labels with a high level of motif-similarity score, and DropNode [12] randomly drop a certain portion of nodes from the original graph and generate a new graph. The random node-dropping method can disconnect closely related nodes, generating isolated nodes in the augmented graph [11]. This may affect the efficiency of the GNN's message-passing mechanism.

Furthermore, the authors of [11] develop a GraphCrop method, which generates various cropped-augmented graphs using a node-centric strategy. However, current graph augmentation methods, such as random node-dropping, often fail to preserve the original graph's essential topological structures during the augmentation process, potentially affecting the performance of the graph classification tasks. Hence, this study aims to tackle this issue by introducing a node-dropping augmentation method. This method removes less important low-degree nodes while preserving the essential topological structures.

## III. Proposed Methodology

This section first defines the mathematical notations and problem formulation of graph classification tasks. Then, we present a detailed description of the proposed NDAUG method. Figure 1 shows the working pipeline of the NDAUG method for generating an augmentation graph.

### A. Mathematical Notations and Problem Formulation

We represent the input graph as $G = (V, E)$, where $V = \{v_i | i = 1, ..., n\}$ denotes the nodes and $E = \{e_i | i = 1, ..., m\}$ shows the edges. The topological structure of $G$ can be represented by an adjacency matrix $A \in \{0, 1\}^{n \times n}$ with $A_{ij} = 1$ if $(i, j) \in E$ and $A_{i,j} = 0$ otherwise. Let $X \in \mathbb{R}^{n \times d}$ represent the node features matrix, where $d$ is the dimension of the feature space. For the graph classification tasks, each data point in the dataset $D = \{(G_i, y_i) | i = 1, ..., t\}$ consists of a graph $G_i$ and its corresponding label $y_i$. We split the dataset $D$ into training, validation, and testing sets, depicted as $D_{train}$, $D_{val}$, and $D_{test}$, respectively (for more detail, see the experimental section IV-A). Specifically, we aim to generate new data samples for a classifier such as $G \in D_{train}$ to a new augmented graph $G'$ like $f : (G, y) \mapsto (G', y)$ where $y$ is the label of $G$. The augmented set $D'_{train}$ add with $D_{train}$ to produce the final training set: $D^{new}_{train} = D_{train} + D'_{train}$.

### B. Motifs Preservation

Motifs are small, recurrent, and connected subgraphs that play a crucial role in measuring the connectivity patterns of nodes within a graph [23]. The importance of motifs in the analysis of graphs has been widely recognized since they serve as key indicators in revealing the fundamental structure and functionality of complex networks [24]. Figure 1 illustrates the topological structures of two benzene rings with 5 and 6 nodes, which we refer to as cyclic motif structures, and these hold substantial chemical significance in molecular datasets. These cyclic motif structures often determine the structural and chemical properties of the molecule, impacting its behaviour and interactions [25]. Therefore, preserving such motifs is critical in graph classification like drug discovery or toxicity prediction, where losing these structures could lead to inaccurate interpretations of the molecule's properties. Current augmentation techniques, like DropNode [12], employ a random process to drop a substantial number of nodes from the graph without preserving the underlying graph motif structures. Therefore, our node-dropping method first preserves these essential topological motif structures during augmentation. Let $M(v)$ be an indicator function defined as:

$$M(v) = \begin{cases} 1 & \text{if } v \in M, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

A node v is preserved in the augmented graph if $M(v) = 1$. Here, $M$ denotes the set of nodes that are part of identified motifs. We implement this $M(v)$ indicator function using the cycle-basis function of the Netwrokx library to identify all cyclic motif structures of nodes 4, 5, and 6.

**Algorithm 1** Node-Dropping Augmentation NDAUG

**Input:** A graph $G$ with vertices $V$ and edges $E$ and Drop Probability $P$
**Output:** Augmented graph $G'$

```
Initialize an empty set D to store nodes to be dropped.
Identify the essential topological structural motifs M in G.
for each vertex v in V do
    Calculate node degree deg(v) = |{e ∈ E : v ∈ e}|
    Generate a random drop probability prob_v for node v
    if d(v) ≤ T_d and prob_v < P and v ∉ M then
        Add v to D.
    end if
end for
Remove the nodes in D from V to get V'.
Update the set of edges E' = {e ∈ E :/ ∃v ∈ D such that v ∈ e}
Re-index nodes in V' starting from 1 up to the size of V'.
return G' = (V', E')
```

### C. Node Degree-based Dropping

After preserving significant structural motifs in the graph, we use the concept of node degree to select the most important nodes to generate an augmented graph. In simple terms, the nodes with a high degree indicate their significance within the entire graph [14]. For example, high-degree nodes in road networks often correspond to major intersections or hub areas [14]. Similarly, in social and protein-protein interaction networks, high-degree nodes often represent influential

TABLE I: Comparison of NDAUG and baselines. The bold text represents the best performances.

| **Methods** | **BZR** | **COX2** | **NCI1** | **MUTAGEN** | **PROTEINS** | **DD** | **IMDB-B** | **RED12K** |
|---|---|---|---|---|---|---|---|---|
| No Augmentation | $79.42_{\pm 1.97}$ | $79.50_{\pm 1.50}$ | $75.50_{\pm 1.50}$ | $77.18_{\pm 1.86}$ | $72.60_{\pm 3.92}$ | $76.50_{\pm 2.35}$ | $68.20_{\pm 6.55}$ | $38.80_{\pm 3.12}$ |
| NodeDrop[12] | $80.31_{\pm 6.50}$ | $78.39_{\pm 3.98}$ | $76.47_{\pm 2.03}$ | $77.77_{\pm 2.57}$ | $73.86_{\pm 2.51}$ | $76.66_{\pm 3.89}$ | $68.20_{\pm 4.98}$ | $41.13_{\pm 1.46}$ |
| EdgeDrop[5] | $81.97_{\pm 3.50}$ | $79.88_{\pm 6.44}$ | $77.93_{\pm 1.33}$ | $79.18_{\pm 1.89}$ | $73.41_{\pm 4.45}$ | $74.03_{\pm 4.09}$ | $69.40_{\pm 4.20}$ | $40.53_{\pm 2.61}$ |
| GraphCrop[11] | $79.84_{\pm 3.40}$ | $79.76_{\pm 4.64}$ | $77.67_{\pm 2.50}$ | $79.54_{\pm 2.59}$ | $73.10_{\pm 3.50}$ | $76.86_{\pm 3.46}$ | $70.87_{\pm 3.51}$ | $40.81_{\pm 2.71}$ |
| Gmixup[29] | $82.15_{\pm 4.25}$ | $78.34_{\pm 5.20}$ | $77.18_{\pm 1.56}$ | $80.59_{\pm 2.31}$ | $72.10_{\pm 5.71}$ | $75.29_{\pm 1.69}$ | $70.31_{\pm 3.36}$ | $41.10_{\pm 2.31}$ |
| M-evolve [22] | $79.30_{\pm 1.53}$ | $77.74_{\pm 3.41}$ | $77.37_{\pm 2.86}$ | $78.84_{\pm 2.25}$ | $72.31_{\pm 3.62}$ | $76.81_{\pm 2.34}$ | $69.40_{\pm 3.81}$ | $40.34_{\pm 3.87}$ |
| Mixup [21] | $81.20_{\pm 3.51}$ | $79.81_{\pm 4.41}$ | $77.08_{\pm 2.10}$ | $79.81_{\pm 2.13}$ | $74.10_{\pm 3.35}$ | $75.40_{\pm 2.80}$ | $69.30_{\pm 3.20}$ | $40.66_{\pm 2.17}$ |
| PiNGDA [21] | $82.35_{\pm 0.51}$ | $79.67_{\pm 0.41}$ | $69.35_{\pm 0.63}$ | $77.21_{\pm 0.13}$ | $73.21_{\pm 0.40}$ | $75.40_{\pm 0.57}$ | $69.21_{\pm 0.20}$ | $41.34_{\pm 0.17}$ |
| **NDAUG** | $\mathbf{86.16}_{\pm 3.21}$ | $\mathbf{81.28}_{\pm 3.37}$ | $\mathbf{80.01}_{\pm 2.51}$ | $\mathbf{82.01}_{\pm 2.21}$ | $\mathbf{75.65}_{\pm 2.54}$ | $\mathbf{79.31}_{\pm 3.31}$ | $\mathbf{72.40}_{\pm 3.20}$ | $\mathbf{45.95}_{\pm 1.61}$ |

individuals or key connectors within the community and correspond to crucial proteins that interact with many other proteins. So, this step removes low-degree nodes from the graph to generate an augmented graph. The degree of a node $d(v)$, which corresponds to the number of edges connecting to $v$, can be formally expressed as:

$$d(v) = |\{u \in V : (u, v) \in E \vee (v, u) \in E\}| \tag{2}$$

The following conditions guide the node-dropping process: nodes are considered to be removed if their degree is less than or equal to the degree threshold $T_d$, their associated drop probability $\text{prob}_v$ is below a threshold $p$, and they are not part of any identified motifs $M$:

$$V_{\text{drop}} = \{v \in V : d(v) \leq T_d \text{ and } \text{prob}_v < p \text{ and } v \notin M\} \tag{3}$$

The augmented graph $G' = (V', E')$ is then derived by removing $V_{\text{drop}}$ from $V$, and accordingly adjusting $E$ to exclude edges incident to any node in $V_{\text{drop}}$. See Algorithm 1 for more details. Formally, the augmented graph is constructed as follows:

$$V' = V \setminus V_{\text{drop}}; \quad E' = \{(u, v) \in E : u \notin V_{\text{drop}} \wedge v \notin V_{\text{drop}}\} \tag{4}$$

### *D. Structure Learning Method*

This section explains the working pipeline of our proposed structure learning mechanism, which learns a refined augmented graph structure. As mentioned earlier, the node-dropping augmentation operation can lead to the disconnection of closely related nodes in the augmented graph $G' = (V', E')$. This loss of graph structure information further hinders the message-passing procedure of the GNN [11] (also see an example in Figure. 1). We used a Graph Attention Layer (GAT), which takes structural information $A'$ and hidden representation $X'$ as input to transform node features by aggregating information from their neighborhoods and resulting attention score vector for each node [26] (as shown in the equation. 5). We use cosine similarity to determine the similarity between the transformed features of isolated nodes and those in the main graph. For any isolated node $i$ and a non-isolated node $j$, their similarity is calculated as follows:

$$F' = \text{GAT}(X', A'); \quad S_{ij} = \frac{F'_i \cdot F'_j}{\|F'_i\| \|F'_j\|} \tag{5}$$

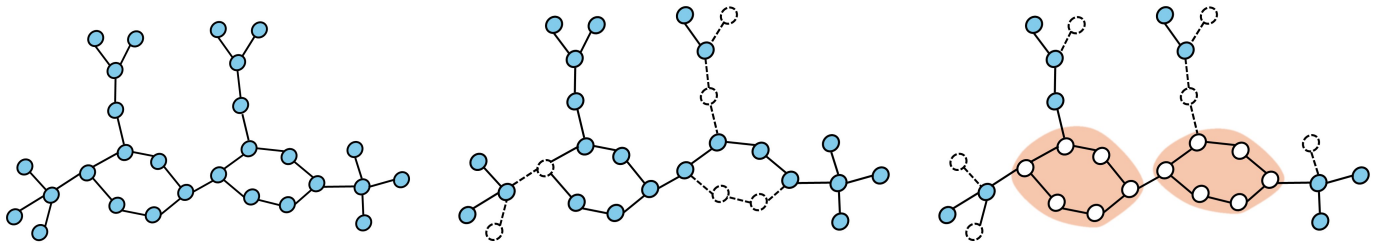
The process of reconnecting isolated nodes involves identifying these nodes, represented as the set $I \subseteq V'$ in $G'$. For each isolated node $i \in I$, we search within $V' \setminus I$ to find a node $j$ that has the highest cosine similarity score with $i$ and then create a new edge between nodes $i$ and $j$.

$$E' = E' \cup \{(i, j) : i \in I, j = \underset{k \in \{V' - I\}}{\text{argmax}} S_{ik}, S_{ij}\} \tag{6}$$

By effectively reconnecting isolated nodes, we maintain the graph's connectivity and ensure the continuity and efficacy of the message-passing mechanisms in the GNN, which are essential for accurate graph classification tasks.

## IV. Experiments

This section evaluates the efficacy of the proposed NDAUG method on eight classification datasets, including BZR, COX2, NCI1, and MUTAGENICITY for molecular compound classification, DD and PROTEINS for protein categorization, IMDB-M, and REDDIT-MULTI12K for social network classification [27]. These datasets have been widely used as benchmarks for graph classification tasks, as demonstrated in this study [28]. Our findings demonstrate that NDAUG consistently outperforms the existing baseline approaches. Furthermore, a graph visualization comparison and a comprehensive series of ablation studies are conducted to evaluate the individual



(a) Training on the NCI1 dataset without any augmentation method obtains an average testing accuracy of 78.44±2.71.

(b) The random node-dropping [12] method training achieves an average testing accuracy of 76.47±2.03.

(c) Training with our proposed NDAUG method obtains an average testing accuracy of 80.01±2.51.

Fig. 2: A comparison of NDAUG and the random node-dropping method on the NCI1 dataset shows significant differences. Figure (a) displays a random NCI1 graph with crucial cyclic carbon structures. Figures (b) and (c) illustrate augmented graphs generated by NDAUG and Nodedrop [12], respectively. The random DropNode degrades classification performance by dropping key nodes from carbon structures. In contrast, NDAUG preserves these structures and enhances classification performance.

contributions of different components inside the NDAUG method.

### *A. Baseline Methods and Experimental Settings*

We follow numerous prior research studies [12], [5] and employ the 10-fold cross-validation method, dividing the datasets into training, validation, and testing sets with ratios of 80%, 10%, and 10%, respectively. We report the average test accuracy over ten different runs. The training process utilizes the early-stop mechanism, which terminates when the loss value of the validation set does not decrease for 50 consecutive epochs. We fine-tune hyperparameters for all models on each dataset within the specified range, as follows: 1) initial learning rate $\in \{0.01, 0.0005\}$, 2) embedding dimensions $\in \{64, 128\}$, 3) batch size $\in \{32, 64, 128\}$, 4) DropEdge and DropNode drop ratio $\in \{20\%, 40\%\}$, 5) node degree value $\in \{1, 2, 3\}$ and number of GNN layers $\in \{2, 3, 4\}$. We utilize the Adam optimizer to initialize our model and apply a negative log-likelihood loss function for training. We compare our NDAUG methods, which do not use any data augmentations, and seven graph augmentation baseline methods, including DropNode [12], DropEdge [5], GraphCrop [11], Gmixup [29], M-evolve [22], Mixup [21] and PiNGDA [21]. We use the same GNN model [15] and hyperparameter setting for NDAUG and all baseline augmentation approaches to ensure a fair comparison.

### *B. Performance Comparison*

Table I compares the performance of our proposed NDAUG and baseline methods across the eight graph classification benchmark datasets. Significantly, NDAUG demonstrates superior performance over all baseline methods in all datasets. Specifically, when comparing with the GCN baseline, NDAUG shows a relative accuracy improvement on the BZR, NCI, MUTAGENICITY, DD, and REDIT12K datasets by margins of 6.74%, 4.51%, 4.83%, 2.45%, and 7.15%, respectively. This advancement underscores the efficacy of graph data augmentation in enhancing GNN performance for graph classification tasks. Moreover, NDAUG consistently surpasses traditional augmentation methods like NodeDrop and EdgeDrop. In the realm of chemical molecule datasets, NDAUG outperforms these baselines by an average of 2-4% in BZR, NCI1, and MUTAGENICITY and by 1.4% in COX2. Across the biological and social network datasets, NDAUG achieves an average improvement of 2.0%, 4.0%, respectively. Existing augmentation methods, such as NodeDrop and EdgeDrop, have limited performance because they randomly drop nodes or edges without preserving the connectivity between the nodes in the augmented graph. This destroys the original graph's essential topological structures, leading to the loss of essential label-related information. We additionally provide graph visualizations to represent the effect of different augmentation techniques in Figure 2. Our analysis, supported by the success of NDAUG on graph datasets, validates the effectiveness of our proposed NDAUG method. This advancement not only sets a new standard in graph augmentation but also opens the potential for future analyses to enhance the performance of GNNs. The overall time complexity of NDAUG, which depends on determining motifs and calculating pairwise similarity, is $O(|V|^c + |V|^2 d)$, where $|V|$ is the number of nodes, $d$ is the dimension of node features, and $c$ represents the complexity of detecting cyclic motifs.

### *C. Ablation Study*

This section performs an ablation study on NDAUG by removing three components to verify further where the performance improvement comes from. For convenience, we name the NDAUG method without the node degree measurement, motif structures, and structure learning components as NDAUG w/o NDM, NDAUG w/o MS, and NDAUG w/o ST, respectively. For ablation study experiments, we train GCN-based [15] classification models on four different-scale graph datasets covering small and large graphs and employ the same parameter setting as Section IV-A. The results presented in Table II highlight the considerable impact of node degree measurement and motif structures, particularly within the domains of chemical molecules, since the preservation of essential motif graph structures such as cyclic benzene with node degree is especially useful to maintain the important

TABLE II: Results of ablation studies about different NDAUG components.

| Architecture | Mutagenicity | NCI1 | BZR | IMDB-B |
|---|---|---|---|---|
| NDAUG | **82.021 ± 2.21** | **80.34 ± 2.39** | **86.16 ± 3.21** | **72.70 ± 2.71** |
| NDAUG w/o ST | 80.90 ± 2.27 | 79.10 ± 2.30 | 84.31 ± 3.11 | 71.40 ± 4.01 |
| NDAUG w/o MS | 80.30 ± 2.96 | 79.87 ± 2.10 | 84.41 ± 3.01 | 70.80 ± 3.61 |
| NDAUG w/o NDM | 78.95 ± 3.01 | 77.83 ± 3.03 | 81.61 ± 3.51 | 70.80 ± 3.61 |

graph structures within the augmented graphs. Furthermore, removing the structure learning strategy significantly degrades the performance of NDAUG in NCI and BZR because these datasets are sparse, resulting in augmented graphs containing isolated nodes. It is demonstrated that structure learning, node degree measurement, and preservation of essential graph motif structures are key success factors of NDAUG in generating augmented graphs.

## V. Conclusions and Future Work

This paper introduced a novel data augmentation method named NDAUG for graph classification tasks. At its core, NDAUG used the concept of node-degree measurement to strategically drop less important low-degree nodes from the original graph. This approach is carefully balanced to maintain essential topological motif structures within the augmented graph, even those typically associated with low-degree nodes. Furthermore, we proposed a structure learning technique that employs an attention mechanism to reconnect disconnected nodes to maintain graph connectivity within the augmented graphs and enhance GNN performance. Comprehensive experiments on eight graph classification datasets demonstrated a notable enhancement in the accuracy of up to $5\%$ compared to the existing baselines. In future work, we plan to enhance our NDAUG approach by incorporating edge perturbation techniques to identify key topological substructures and extend its application to node-level tasks.

## References


[1] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

[2] Kim, M., Choi, J., Lee, S., Jung, J., & Kang, U. (2025). Augward: Augmentation-aware representation learning for accurate graph classification. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 237-250). Singapore: Springer Nature Singapore.

[3] Perez, L., & Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. arXiv preprint arXiv:1712.04621.

[4] Liu, X., Zhang, Y., Wu, M., Yan, M., He, K., Yan, W., & Fan, D. (2025). Revisiting edge perturbation for graph neural network in graph data augmentation and attack. IEEE Transactions on Knowledge and Data Engineering.

[5] Rong, Y., Huang, W., Xu, T., & Huang, J. (2019). Dropedge: Towards deep graph convolutional networks on node classification. arXiv preprint arXiv:1907.10903.

[6] Liu, S., Ying, R., Dong, H., Li, L., Xu, T., Rong, Y., & Wu, D. (2022). Local augmentation for graph neural networks. In International conference on machine learning (pp. 14054-14072). PMLR.

[7] Tuggener, L., Emberger, R., Ghosh, A., Sager, P., Satyawan, Y. P., Montoya, J., Goldschagg, S., Seibold, F., Gut, U., Ackermann, P. Schmidhuber, J., & Stadelmann, T. (2024). Real world music object recognition. Transactions of the International Society for Music Information Retrieval, 7(1), 1-14.

[8] Huang, S., Xu, Y., Zhang, H., & Li, X. (2025). Learn beneficial noise as graph augmentation. arXiv preprint arXiv:2505.19024.

[9] Verma, V., Qu, M., Kawaguchi, K., Lamb, A., Bengio, Y., Kannala, J., & Tang, J. (2021). Graphmix: Improved training of gnns for semi-supervised learning. In Proceedings of the AAAI conference on artificial intelligence (pp. 10024-10032).

[10] G Taylor Kong, K., Li, G., Wu, Z., Zhu, C., Ghanem, B., Taylor, G. & Goldstein, T. (2020). Flag: Adversarial data augmentation for graph neural networks.

[11] Wang, Y., Wang, W., Liang, Y., Cai, Y., & Hooi, B. (2020). Graphcrop: Subgraph cropping for graph classification. arXiv preprint arXiv:2009.10564.

[12] You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., & Shen, Y. (2020). Graph contrastive learning with augmentations. Advances in neural information processing systems.

[13] Sui, Y., Wang, S., Sun, J., Liu, Z., Cui, Q., Li, L., & He, X. (2025). A simple data augmentation for graph classification: A perspective of equivariance and invariance. ACM Transactions on Knowledge Discovery from Data, 19(2), 1-24.

[14] Song, C., Lin, Y., Guo, S., & Wan, H. (2020). Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In Proceedings of AAAI conference on artificial intelligence (pp. 914-921).

[15] Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., & Grohe, M. (2019). Weisfeiler and leman go neural: Higher-order graph neural networks. In Proceedings of AAAI conference on artificial intelligence (4602-4609).

[16] Kang, J., Liu, S., Huang, K., Fan, C., He, H., & Chen, C. (2024). Graph-CoRe: Graph Representation Learning with Contrastive Subgraph Replacement. In IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (pp. 511-518). IEEE.

[17] Zhou, J., Xie, C., Gong, S., Wen, Z., Zhao, X., Xuan, Q., & Yang, X. (2025). ACM Computing Surveys, 57(11), 1-34.

[18] Wang, Y., Zhao, Y., Shah, N., & Derr, T. (2022). Imbalanced graph classification via graph-of-graph neural networks. In Proceedings of the 31st ACM international conference on information & knowledge management (pp. 2067-2076).

[19] Zhao, T., Jin, W., Liu, Y., Wang, Y., Liu, G., Günnemann, S.,& Jiang, M. (2022). Graph data augmentation for graph machine learning: A survey.

[20] Wang, Z., Yin, Z., Zhang, Y., Yang, L., Zhang, T., Pissinou, N., & Zhang, W. (2025). Fg-smote: Towards fair node classification with graph neural network. ACM SIGKDD Explorations Newsletter, 26(2), 99-108.

[21] Wang, Y., Wang, W., Liang, Y., Cai, Y., & Hooi, B. (2021). Mixup for node and graph classification. In Proceedings of the Web Conference (pp. 3663-3674).

[22] Zhou, J., Shen, J., Yu, S., Chen, G., & Xuan, Q. (2020). M-evolve: structural-mapping-based data augmentation for graph classification. IEEE Transactions on Network Science and Engineering, 8(1), 190-200.

[23] Elhesha, R., & Kahveci, T. (2016). Identification of large disjoint motifs in biological networks. BMC bioinformatics, 17, 1-18.

[24] Sun, M., Yang, M., Li, Y., Mu, D., Wang, X., & Wang, Y. (2023). Structural-aware motif-based prompt tuning for graph clustering. Information Sciences, 649, 119643.

[25] Meyer, E. A., Castellano, R. K., & Diederich, F. (2003). Interactions with aromatic rings in chemical and biological recognition. Angewandte Chemie International.

[26] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.

[27] Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., & Neumann, M. (2020). Tudataset: A collection of benchmark datasets for learning with graphs. arXiv preprint arXiv:2007.08663.

[28] Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2018). How powerful are graph neural networks?. arXiv preprint arXiv:1810.00826.

[29] Han, X., Jiang, Z., Liu, N., & Hu, X. (2022, June). G-mixup: Graph data augmentation for graph classification. In International Conference on Machine Learning.