ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240652

Enhancing Node Representations for Real-World Complex Networks with Topological Augmentation

Xiangyu Zhao[a,*,1], **Zehui Li**[a,*,1], **Mingzhu Shen**[a], **Guy-Bart Stan**[a], **Pietro Liò**[b] **and Yiren Zhao**[a]

[a]Imperial College London, United Kingdom
[b]University of Cambridge, United Kingdom

Abstract. Graph augmentation methods play a crucial role in improving the performance and enhancing generalisation capabilities in Graph Neural Networks (GNNs). Existing graph augmentation methods mainly perturb the graph structures, and are usually limited to pairwise node relations. These methods cannot fully address the complexities of real-world large-scale networks, which often involve higher-order node relations beyond only being pairwise. Meanwhile, real-world graph datasets are predominantly modelled as simple graphs, due to the scarcity of data that can be used to form higher-order edges. Therefore, reconfiguring the higher-order edges as an integration into graph augmentation strategies lights up a promising research path to address the aforementioned issues. In this paper, we present *Topological Augmentation (TopoAug)*, a novel graph augmentation method that builds a combinatorial complex from the original graph by constructing virtual hyperedges directly from the raw data. TopoAug then produces auxiliary node features by extracting information from the combinatorial complex, which are used for enhancing GNN performances on downstream tasks. We design three diverse virtual hyperedge construction strategies to accompany the construction of combinatorial complexes: (1) via graph statistics, (2) from multiple data perspectives, and (3) utilising multi-modality. Furthermore, to facilitate TopoAug evaluation, we provide 23 novel real-world graph datasets across various domains including social media, biology, and e-commerce. Our empirical study shows that TopoAug consistently and significantly outperforms GNN baselines and other graph augmentation methods, across a variety of application contexts, which clearly indicates that it can effectively incorporate higher-order node relations into the graph augmentation for real-world complex networks.

1 Introduction

Graph Neural Networks (GNNs) [3, 13, 20, 33, 41] are powerful tools for learning the representation of relationships between objects, ranging from social networks [25], biology [29] to e-commerce [16, 24, 26]. Representation learning tasks, such as node prediction, constitute a major category of tasks for GNNs. However, limitations in the generalisability of GNNs have obstructed their broader application in real-world settings [30].

Several graph augmentation techniques have been introduced to enhance the performance and foster better generalisation for GNNs, specifically in the context of node prediction tasks involving large graphs. Current graph augmentation methods include graph structures perturbation [6, 28], feature perturbation [9], and label-oriented augmentations [15, 35]. Simple-GNNs – GNNs that operate on simple graphs with only pairwise relations – normally use perturbation-based augmentation techniques. However, these augmentation methods do not adequately grasp the complexities of real-world, large-scale networks that frequently display intricate node relationships beyond having only pairwise connections. While higher-order GNNs, such as Hypergraph Neural Networks (hyper-GNNs) [7, 34, 39], have been developed to model higher-order node relations, they encounter a significant challenge due to the scarcity of explicitly recorded data for forming high-order edges in many real-world datasets. This scarcity hampers the direct application of hyper-GNNs, as they rely on comprehensive data that capture complex node relations.

Although the scarcity of higher-order edges presents a challenge, integrating advanced modeling methods into augmentation strategies offers a promising avenue to address this issue. Traditionally, graph augmentation has focused on first-order connections, primarily modifying or adding direct links between nodes [6, 28]. However, this approach overlooks the rich, multi-level interactions captured in higher-order edges, which encompass indirect connections, such as patterns of group interactions within the networks. By incorporating these higher-order relations, graph augmentation can achieve a more nuanced understanding of network dynamics, revealing hidden structures and patterns that are not apparent in direct links alone. This approach is particularly beneficial in real-world complex networks, where the interplay of various types of connections can provide deeper insights into the underlying system, across a wide range of domains including social media, biological ecosystems and e-commerce networks. Therefore, the integration of higher-order edge information into graph augmentation strategies opens up new possibilities for more sophisticated and accurate network analysis.

In our study, we introduce a suite of graph augmentation techniques that extend beyond conventional approaches. Our methodology encompasses higher-order node relations, and we present a novel graph augmentation strategy termed *Topological Augmentation (TopoAug)*. As illustrated in Figure 1, TopoAug employs a two-step augmentation mechanism. First, TopoAug constructs hyperedges from the original graph data, thereby forming a *combinatorial complex* [12]. Second, TopoAug further processes the combinatorial complex through an auxiliary function (f_{aux} in Figure 1) to produce auxiliary node embeddings. These auxiliary embeddings can then be used by GNNs to improve their robustness and accuracy on downstream tasks. We present three distinct ways for constructing these hyperedges: (1) via graph statistics, (2) from multiple data perspectives, and (3) using mul-

* Equal contribution.
1 Corresponding authors. Email: {x.zhao22,zehui.li22}@imperial.ac.uk.

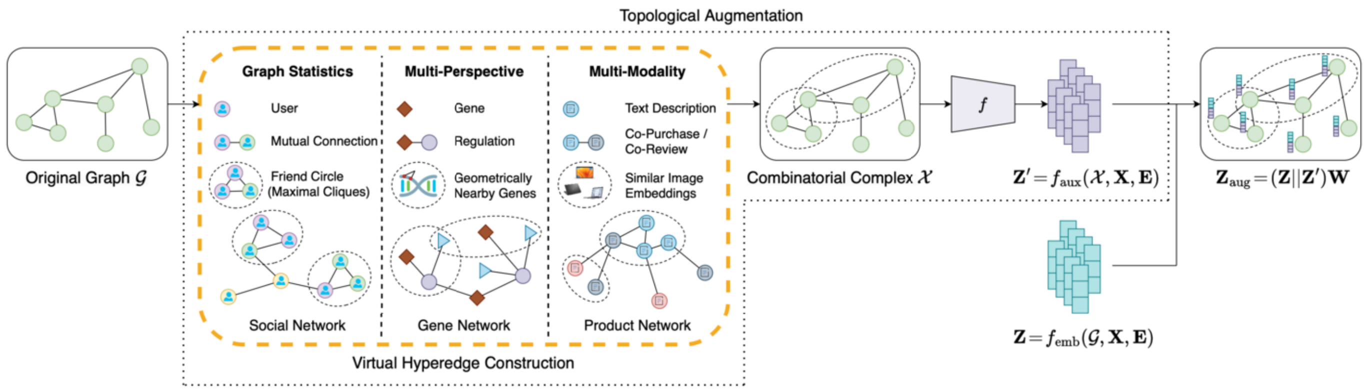

Topological Augmentation
Original Graph $\mathcal{G}$
Graph Statistics
User
Mutual Connection
Friend Circle (Maximal Cliques)
Social Network
Multi-Perspective
Gene
Regulation
Geometrically Nearby Genes
Gene Network
Multi-Modality
Text Description
Co-Purchase / Co-Review
Similar Image Embeddings
Product Network
Virtual Hyperedge Construction
Combinatorial Complex $\mathcal{X}$
f
$\mathbf{Z}' = f_{\text{aux}}(\mathcal{X}, \mathbf{X}, \mathbf{E})$
$\mathbf{Z}_{\text{aug}} = (\mathbf{Z}||\mathbf{Z}')\mathbf{W}$
$\mathbf{Z} = f_{\text{emb}}(\mathcal{G}, \mathbf{X}, \mathbf{E})$

Figure 1: An overview of the TopoAug method showcasing three virtual hyperedge construction strategies: via graph statistics, from multiple data perspectives, and using multiple data modalities. The process initiates with an original graph $\mathcal{G}$. The virtual hyperedge construction methods contribute to the construction of an combinatorial complex $\mathcal{X}$, which is then processed through an auxiliary function f_{aux} to obtain a set of auxiliary node features $\mathbf{Z}'$. The auxiliary features $\mathbf{Z}'$ are then combined with the unaugmented node embeddings $\mathbf{Z}$ (obtained from the original graph via the backbone embedding function f_{emb}), to produce the final augmented node embeddings $\mathbf{Z}_{\text{aug}}$. The final augmented node embeddings $\mathbf{Z}_{\text{aug}}$ are then used in downstream tasks to enhance prediction accuracy.

tiple data modalities. Our augmentation schemes showcase notable flexibility across a variety of datasets, catering to diverse applications such as social, biological, e-commerce, and knowledge networks. Our main contributions are as follows:

- We introduce TopoAug, a novel data augmentation method for GNNs on node prediction tasks. TopoAug allows the network to integrate high-order edge information. Along with our method, we also release 23 new datasets, encompassing diverse domains including social media, biology as well as e-commerce networks.
- We assess various design choices within TopoAug, including different methods for constructing virtual hyperedges and varying auxiliary functions. Our examination of diverse TopoAug variants across multiple GNNs and prediction tasks underscores TopoAug's versatility for various application contexts.
- Through a comprehensive empirical evaluation on various real-world datasets and GNN architectures, we demonstrate that TopoAug offers consistent and significant performance improvements over GNN baselines and existing graph augmentation methods.

2 Related Work

Topological Deep Learning Topological deep learning extends machine learning models from graph data to data on topological domains, such as hypergraphs and other higher-order graphs, to facilitate the exponential growth in both the amount and the complexity of data for computational analysis. Formally, a (simple) graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a collection of nodes $\mathcal{V}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ between pairs of nodes. This simple graph abstraction assumes that each edge only connects two nodes. However, as discussed in the previous sections, many real-world networks have more complex node relations than just pairwise relations. A hypergraph is also defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, but $\mathcal{E} \subseteq \mathcal{P}(\mathcal{V}) \setminus \varnothing$ is now the set of hyperedges, each of which can connect two or more nodes. However, as real-world datasets are predominantly recorded as simple graphs, treating both simple edges and "true" hyperedges (edges that connects strictly more than two nodes) as hyperedges altogether also limits their flexibility, due to the scarcity of data that can form true hyperedges. Recently, there emerges a novel type of topological domain – *combinatorial complexes*, that enrich the hypergraphs by separating simple edges and true hyperedges through the inclusion of hierarchical ranks [12]:

Definition 1 (Combinatorial complex)**.** *A* ***combinatorial complex*** *is a triple* $(\mathcal{V}, \mathcal{X}, \mathrm{rk})$ *consisting of a node set* $\mathcal{V}$*, a node relation set* $\mathcal{X} \subseteq \mathcal{P}(\mathcal{V}) \setminus \varnothing$*, and a rank function* $\mathrm{rk} : \mathcal{X} \to \mathbb{N}$*, such that*

1. $\forall v \in \mathcal{V}.\ \{v\} \in \mathcal{X}$*; and*
2. *the rank function* rk *is order-preserving, which means that* $\forall x, y \in \mathcal{X}.\ x \subseteq y \implies \mathrm{rk}(x) \leq \mathrm{rk}(y)$.

For brevity, $\mathcal{X}$ is used as an abbreviated notation for a combinatorial complex $(\mathcal{V}, \mathcal{X}, \mathrm{rk})$. Typically, the rank of any singleton node relation $\{v\}$ in $\mathcal{X}$ is set to zero, to make it naturally aligned with simple graphs and hypergraphs. The rank function effectively induces a hierarchical structure on $\mathcal{X}$, which can be used to separate simple edges and true hyperedges, by assigning them with different ranks.

Graph Neural Networks for Simple Graphs Graph Neural Networks (GNNs) on simple graphs encode the nodes through neural networks, and learn the representations of the nodes through message-passing within the graph structure. GCN incorporates the convolution operation in computer vision into GNNs [20]. GAT [33] and GATv2 [3] are another family of GNN variants that focus on improving the expressive power of GNNs by using the attention mechanism. GraphSAGE [13] is a general inductive framework that leverages node information to efficiently generate node embeddings for previously unseen data. GraphSAINT [41] underscores the significance of graph sampling-based inductive learning – it utilises diverse graph sampling techniques and illustrates how learning on smaller, sampled graphs can enhance training efficiency, particularly with large graphs. While these models demonstrate the success on simple graph datasets, they continue to face challenges in their generalisation to unseen data and are susceptible to small variations in graph structures [4].

Data Augmentation Methods for GNNs Enhancing the generalisability of GNNs through graph augmentation methods remains a key area of research interest in recent years. Existing graph augmentation methods introduce perturbations to the graph structure, enabling GNNs trained on these altered graphs to learn and capture invariance. This includes adding perturbations to the graphs' adjacency matrices through DropEdge [28], randomly removing nodes through DropNode [6, 40], and perturbing node edge features through feature masking. Another line of graph augmentations relies on the generation of synthetic data, which can be achieved through the interpolation of existing data via Mixup [15, 35, 42], or using generative models to enrich the training data [21]. However, these existing augmentation strategies do not inherently increase the expressiveness of GNNs. Standard GNNs typically aggregate information from their neighbourhoods, a process that shares limitations with the Weisfeiler-Lehman (1-WL) graph isomorphism test [14, 37, 38]. For instance, as depicted in Figure 2(a), simple-GNNs fail to distinguish between two structurally different graphs with the same node degrees due to this limitation. This research aims to develop augmentation-based methods that address and overcome these constraints, thereby enhancing the capabilities of GNNs beyond their current limits.

Representation Learning on Hypergraphs Hypergraphs are designed to capture more complex node relations, where an edge can connect two or more nodes. In general, GNNs for hypergraphs optimise the node representation through a two-step process. Initially, the node embeddings within each hyperedge are aggregated to form a hidden embedding of each hyperedge. Subsequently, the hidden embeddings of hyperedges with common nodes are aggregated to update the representations of their common nodes. Both HGNN [10] and HyperConv [1] precisely follow this process. The expressiveness of hypergraph GNNs could be enhanced by modifying this procedure. For instance, HyperGCN [39] refines the node aggregation within hyperedges using mediators [5]. HyperAtten [1] uses attention to measure the degree to which a node belongs to a hyperedge. HNHN [7] applies nonlinear functions to both node and edge aggregation processes. ED-HNN [34] approximates continuous equivariant hypergraph diffusion operators on hypergraphs by feeding node representations into the message from hyperedges to nodes. GNNs designed for hypergraphs are effective in modelling complex networks. However, real-world datasets are predominantly recorded as simple graphs, which has limited the application of hyper-GNNs due to the scarcity of data that can form hyperedges. In this study, we propose TopoAug that derives hyperedges directly from the raw data, thereby extending the applicability of hyper-GNNs to cases where only conventional simple graph data are available. By utilising higher-order GNNs on virtually established hyperedges to produce auxiliary features, TopoAug effectively broadens the potential of hyperedges to scenarios previously confined to simple graph data.

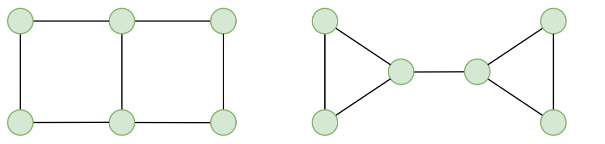
3 Method

3.1 Combinatorial Complex Construction

Inspired by the combinatorial complex modelling, TopoAug attempts to capture complex relations in real-world large networks by constructing a combinatorial complex from the original network. Specifically, TopoAug augments a simple graph by constructing a set of hyperedges $\mathcal{E}_h \subseteq \mathcal{P}(\mathcal{V}) \setminus \varnothing$ from the original graph, via a hyperedge extraction function $h : \mathfrak{G} \times \mathbb{R}^{|\mathcal{V}|\times d_v} \times \mathbb{R}^{|\mathcal{E}|\times d_e} \to \mathcal{P}(\mathcal{P}(\mathcal{V}))$, where $\mathfrak{G}$ denotes the input graph space, $\mathbb{R}^{|\mathcal{V}|\times d_v}$ denotes the d_v-dimensional node feature space, $\mathbb{R}^{|\mathcal{E}|\times d_e}$ denotes the d_e-dimensional edge feature space, and $\mathcal{P}(\mathcal{P}(\mathcal{V}))$ represents the set of all possible collections of hyperedges, which defines the output hyperedge space:

$$\mathcal{E}_h = h(\mathcal{G}, \mathbf{X}, \mathbf{E}) \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}|\times d_v}$ denotes the node feature matrix of the graph, with each row $\mathbf{x}_v \in \mathbb{R}^{d_v}$ being the d_v-dimensional features of node v, and $\mathbf{E} \in \mathbb{R}^{|\mathcal{E}|\times d_e}$ denotes the edge feature matrix of the graph, with each row $\mathbf{e}_e \in \mathbb{R}^{d_e}$ being the d_e-dimensional features of edge e. Each hyperedge $e_h \in \mathcal{E}_h$ is a subset of $\mathcal{V}$ containing at least three nodes, thereby capturing complex multi-node relations. TopoAug then constructs the combinatorial complex $(\mathcal{V}, \mathcal{X}, \mathrm{rk})$ from the original graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and extracted hyperedges $\mathcal{E}_h$, abbreviated as $\mathcal{X}$, according to the following rules:


(a) Without TopoAug

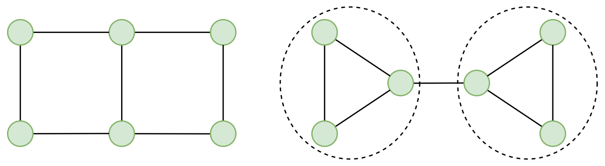

(b) With TopoAug

Figure 2: Graph statistics-based TopoAug can help GNNs surpass the limitations posed by the 1-WL test. (a) Without TopoAug, GNNs are unable to distinguish the two example non-isomorphic graphs that have the same node degrees; (b) Since TopoAug identifies that the right graph contains two cliques, while the left graph contains none, GNNs can now successfully distinguish those graphs.

$$\mathcal{V} \text{ remains unchanged from the original graph} \tag{2}$$

$$\mathcal{X} = \{\{v\} \mid v \in \mathcal{V}\} \cup \{\{u, v\} \mid (u, v) \in \mathcal{E}\} \cup \mathcal{E}_h \tag{3}$$

$$\forall x \in \mathcal{X}.\, \mathrm{rk}(x) = \begin{cases} 0 & \text{for } x = \{v\} \text{ where } v \in \mathcal{V} \\ 1 & \text{for } x = \{u, v\} \text{ where } (u, v) \in \mathcal{E} \\ 2 & \text{otherwise (i.e., for } x \in \mathcal{E}_h) \end{cases} \tag{4}$$

Depending on the nature of the original graph, the virtual hyperedges can be constructed in one of the following three ways:

From the Graph Statistics For graphs that can reveal valuable information from their local clusters and cliques, such as the social networks, TopoAug constructs hyperedges by computing from graph statistics, such as maximal cliques. TopoAug then groups those nodes within the same maximal cliques. This augmentation acts as a complement to the simple edges in the original graph by calculating the local node-wise clustering relationships. TopoAug then enables GNNs seeking to utilise local clustering information to rapidly access that information without needing to explicitly precompute it. In addition, by separating out the local clusters and cliques from the simple edge adjacency of the original graph, TopoAug can prevent GNNs from mixing up those information, thereby potentially enhancing its training performance. Another crucial advantage for this type of augmentation is that it enables GNNs to overcome the expressiveness limitations posed by the 1-WL test [14, 37, 38]: it is able to help GNNs distinguish non-isomorphic graphs with identical node degrees, by specifying their distinct maximal cliques, as illustrated in Figure 2.

From a different Data Perspective Many graphs carry information from different data perspectives. Stacking various types of information from different data perspectives, whether as the graph's node features or edges, can potentially make the learning challenging, since the underlying GNN would then have to learn from these manually mixed information. Real-world examples of such graphs include the biological networks, where both entity-entity interactions and geometrical information are critical to the prediction tasks. For example, gene regulatory interactions might be represented as an adjacency matrix, while the spatial positioning of genes can be used to establish higher-order relationships. The data inherently offer multiple ways for connecting entities within a graph. In this scenario, TopoAug constructs hyperedges by extracting information, preferably featuring multi-node relations, from a different data perspective than the node embeddings and edges. These hyperedges can then be processed separately from the node embeddings and edge adjacency, which are derived from a different data perspective, making them ultimately serve as auxiliary node embeddings for the prediction tasks. This type of virtual hyperedge augmentation serves not only as a complement to the simple edges, but also as an enhancement to the overall information obtained – it takes additional information from a different data source that contain high-order node relations.

From a different Data Modality For graphs that incorporate information from various data modalities, such as product networks containing both textual and image data, effectively integrating the information from different data modalities can be challenging for a GNN. Similar to the previous scenario, TopoAug addresses this by constructing hyperedges using information from one of the data modalities, preferably exhibiting multi-node relations, which are distinct from the other data modalities used by node embeddings and edges. With this approach, information from different data modalities can be processed independently, and the data from the additional modality essentially serve as an augmentation. Note that this differs from simply incorporating an additional data source into the node embedding, as the alternative data modality may only convey 'grouping' information, which is more suitably represented as higher-order graph relations, such as hyperedges. As the hyperedges are constructed from a different modality, this type of hyperedge augmentation provides both a complement to the simple edges of the original graph, and an enhancement to the overall information.

In topological deep learning, while the simple edges of the graph can be represented as an adjacency matrix $\mathbf{A} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$, where $A_{uv} = 1$ if $(u,v) \in \mathcal{E}$ and 0 otherwise, the hyperedges, as well as the final augmented combinatorial complex, can be represented as incidence matrices $\mathbf{H} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{E}_h|}$ and $\mathbf{H} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{X}|}$ respectively, with each entry H_{ve_h} $(H_{vx}) = 1$ if $v \in e_h$ (x) and 0 otherwise. The combinatorial complex $\mathcal{X}$ is then further processed to produce the final auxiliary features for the original graph's nodes.

3.2 Utilising the Hyperedge Information

In order to thoroughly and appropriately utilise the additional information from TopoAug's combinatorial complex, we introduce a node feature augmentation pipeline consisting of:

1. A combinatorial complex construction function $f_{\text{CC}} : \mathfrak{G} \to \mathfrak{X}$ mapping from the graph space $\mathfrak{G}$ to the combinatorial complex space $\mathfrak{X}$. f_{CC} also encapsulates a hyperedge construction function h (defined in Section 3.1); followed by
2. A node embedding function $f_{\text{emb}} : \mathfrak{G} \times \mathbb{R}^{|\mathcal{V}|\times d_v} \times \mathbb{R}^{|\mathcal{E}|\times d_e} \to \mathbb{R}^{|\mathcal{V}|\times d_z}$ that operates on the original graph, together with its unaugmented node and edge features, to compute the d_z-dimensional original node embeddings; and
3. An auxiliary function $f_{\text{aux}} : \mathfrak{X} \times \mathbb{R}^{|\mathcal{V}|\times d_v} \times \mathbb{R}^{|\mathcal{E}|\times d_e} \to \mathbb{R}^{|\mathcal{V}|\times d_{z'}}$ that operates on the constructed combinatorial complex, together with the original graph's unaugmented node and edge features to produce the $d_{z'}$-dimensional auxiliary node features.

The pipeline outputs the final, $d_{z_{\text{aug}}}$-dimensional augmented node embeddings as defined in the following equations:

$$\text{TopoAug}(\mathcal{G}, \mathbf{X}, \mathbf{E}) = (f_{\text{emb}}(\mathcal{G}, \mathbf{X}, \mathbf{E}) \parallel f_{\text{aux}}(f_{\text{CC}}(\mathcal{G}), \mathbf{X}, \mathbf{E}))\mathbf{W} \tag{5}$$

where $\mathbf{W} \in \mathbb{R}^{(d_z+d_{z'})\times d_{z_{\text{aug}}}}$ denotes the weight matrix of the output layer, and $\parallel$ denotes column-wise concatenation. The final augmented node embeddings are then used for the downstream prediction tasks. The complete TopoAug pipeline is illustrated in Figure 1. While this flexible TopoAug pipeline allows other mechanisms to integrate the auxiliary features into the original graph than a simple concatenation followed by an output layer, it is worth pointing out that this work primarily focuses on exploring the potential of leveraging GNN performance through incorporating higher-order node relations into the original graphs. Designing fine-grained feature integration mechanisms is not the main focus of this research.

4 Experiments

4.1 Datasets

To the best of our knowledge, there is a deficiency in the quantity and variety of widely-accepted graph datasets that support data augmentation beyond simple graphs. This is largely attributable to the scarcity of explicitly labelled higher-order node relationships. In order to thoroughly evaluate the effectiveness of TopoAug across varied domains, we build *23 novel graph datasets* derived from real-world networks across varied domains, including social media, biology, and e-commerce. Care has been taken to ensure that the datasets do not contain any personally identifiable information. The 23 datasets can be split into three groups according to the intended virtual hyperedge construction process for TopoAug:

MUSAE We build eight social networks derived from the Facebook pages, GitHub developers and Twitch gamers, plus three English Wikipedia page-page networks on specific topics (chameleons, crocodiles and squirrels) based on MUSAE [29]. Nodes represent users or articles, and edges are mutual followers relationships between the users, or mutual links between the articles. These datasets are intended to assess TopoAug's effectiveness in constructing virtual hyperedges from the graph statistics: on these datasets, TopoAug constructs the virtual hyperedges to be mutually connected sub-groups that contain at least three nodes (i.e., maximal cliques with sizes of at least 3). The tasks for the Facebook, GitHub, and Twitch datasets involve multi-class classification to predict the categories of users or pages, while the task for the Wiki dataset is a regression task that predicts the average monthly traffic of a web page.

GRAND We select and build ten gene regulatory networks in different tissues and diseases from GRAND [2], a public database for gene regulation. Nodes represent gene regulatory elements [23] with three distinct types: protein-encoding gene, lncRNA gene [22], and other regulatory elements. Edges are regulatory effects between genes. We train a CNN [8] and use it to take the gene sequence as input and create a 4,651-dimensional embedding for each node. These datasets are intended to assess TopoAug's effectiveness in constructing virtual hyperedges from a different data perspective: on these datasets, TopoAug constructs the virtual hyperedges by grouping geometrically nearby genomic elements on the chromosomes, i.e., the genomic elements within 200k base pair distance. The task is a multi-class classification of gene regulatory elements.

Amazon Following existing works on graph representation learning on e-commerce networks [31, 41], we faithfully reconstruct a subset of the OGB [17] `ogbn-products` dataset, and build two product co-purchase/co-review networks based on the Amazon Product Reviews dataset [16, 24, 26]. Nodes represent products, and an edge between two products is established if a user buys or writes reviews for both products. Node features are extracted based on the textual description

Table 1: Aggregated statistics of the datasets used for TopoAug evaluation.

Name	Hyperedge Construction Mechanism	#Datasets	Avg. #Nodes	Avg. #Edges	Avg. #Hyperedges	Avg. Node Degree	Avg. Hyperedge Degree	#Classes
MUSAE-GitHub	Graph Statistics	1	37,700	578,006	223,672	30.7	4.6	4
MUSAE-Facebook	Graph Statistics	1	22,470	342,004	236,663	30.4	9.9	4
MUSAE-Twitch	Graph Statistics	6	5,686	143,038	110,142	50.6	6.0	2
MUSAE-Wiki	Graph Statistics	3	6,370	266,998	118,920	88.8	14.4	Regression
GRAND-Tissues	Multi-Perspective	6	5,931	5,926	11,472	2.0	1.3	3
GRAND-Diseases	Multi-Perspective	4	4,596	6,252	7,743	2.7	1.3	3
Cora	Multi-Perspective	2	2,708	5,429	1,326	4.0	3.5	7
Pubmed	Multi-Perspective	1	19,717	44,338	7,963	4.5	4.3	3
Amazon-Computers	Multi-Modality	1	10,226	55,324	10,226	10.8	4.0	10
Amazon-Photos	Multi-Modality	1	6,777	45,306	6,777	13.4	4.8	10

of the products. These datasets are intended to assess TopoAug's effectiveness in constructing virtual hyperedges from a different data modality: on these datasets, TopoAug introduces the image modality into the construction of virtual hyperedges. To be specific, the raw images of the products are fed into a CLIP [27] classifier, and a 512-dimensional feature embedding for each image is returned to assist the clustering. TopoAug then constructs the virtual hyperedges by grouping products whose image embeddings have pairwise distances within a certain threshold. The task is to predict the sub-category of a product in a multi-class classification setup.

In addition to the 23 novel datasets, we also adopt three commonly used citation datasets: Cora-CoCitation, Cora-CoAuthorship, and Pubmed-CoCitation, to match TopoAug's performance with the community standard. We use these three datasets to also assess TopoAug's effectiveness in constructing virtual hyperedges from a different data perspective: the edges of the original graph and the virtual hyperedges constructed by TopoAug are co-citation links and co-authorship groups respectively, or vice versa (i.e., co-authorship links and co-citation groups). Table 1 reports the key graph statistics for each dataset group, and more details of the datasets are described in [43]. We make our source code and full datasets publicly available at https://github.com/VictorZXY/TopoAug.

4.2 Experimental Setup

Training Details We run all the experiments on NVIDIA A100 and V100 GPUs, with up to 40GB memory. Adam [19] is used as the optimiser, and CosineAnnealingLR [11] is used as the learning rate scheduler for all training. For each experiment, the nodes of the graph dataset are randomly split into training, validation, and test sets with a split ratio of 6:2:2. All models are trained for 500 epochs. For node classification tasks, the negative log likelihood loss (NLLLoss) is used as the loss function. For node regression tasks, the mean square error (MSELoss) is used as the loss function. Each experiment typically takes less than 5 minutes to train, when ED-HNN is not incorporated in the model. Due to the significantly larger architecture of ED-HNN, experiments involving it can take up to 2 hours.

Hyperparameter Settings We perform a hyperparameter search for the learning rate and dropout rate, while keeping the hidden dimension of the layers fixed as 64. To ensure fair comparison, all evaluated GNNs (GCN, GAT, GraphSAGE, HyperConv, and ED-HNN) share the same hyperparameter combinations. After hyperparameter searching, we adopt the following hyperparameter selections: learning rate = 0.001, and dropout rate = 0.5. For the additional hyperparameters of ED-HNN, we closely adhere to the hyperparameter settings specified in the original ED-HNN paper [34], and set the number of all inner multi-layer perceptrons (MLPs) within ED-HNN to 2.

4.3 Designing TopoAug

As described in Section 3.2, the TopoAug pipeline consists of a combinatorial complex construction function f_{CC} that constructs a combinatorial complex from the original graph, followed by a backbone node embedding function f_{emb} that operates on the original graph to produce the unaugmented node embeddings, and an auxiliary function f_{emb} that operates on the constructed combinatorial complex to produce the auxiliary node features. This opens up the following questions regarding the practical design of TopoAug:

- At what point should the auxiliary node features be integrated with the original node information – directly at the input side, or within the middle-layer embeddings?
- What is the appropriate type, and consequently, the optimal choice of the auxiliary function?

We investigate these design choices and analyse the optimal usage of TopoAug through a series of ablation studies.

4.3.1 Appropriate Phase for Applying TopoAug

The auxiliary features generated by TopoAug provide significant flexibility regarding the stage of integration, including simultaneous injection with the input node features or blending with the original GNN's activations. To thoroughly identify the appropriate place for inserting TopoAug's auxiliary features, we compare the performance of TopoAug with the following settings:

1. concatenate TopoAug's auxiliary features directly with the input node features, then proceed the concatenated features with a linear layer, before feeding into the embedding GNN (denoted as $\mathrm{TopoAug}_{(f_{\mathrm{aux}},\,\mathrm{input})}$, where f_{aux} is the auxiliary function);
2. concatenate TopoAug's auxiliary features with the GNN activations, followed by a linear layer (denoted as $\mathrm{TopoAug}_{(f_{\mathrm{aux}},\,\mathrm{emb.})}$).

We then conducted experiments on the MUSAE-GitHub, GRAND-Brain and Amazon-Computers datasets, each featuring a distinct virtual hyperedge construction strategy, to evaluate these two settings. The augmentations are applied to a GCN [20]. HyperConv [1] and ED-HNN [34] are selected as the auxiliary functions for TopoAug, since HyperConv is one of the most popular hyper-GNN baselines, and ED-HNN represents the current state-of-the-art in hyper-GNNs.

Table 2 summarises the results for this set of ablation studies. The results show a consistent and significant increase in accuracy for GCN with $\mathrm{TopoAug}_{(f_{\mathrm{aux}},\,\mathrm{emb.})}$ compared to the vanilla GCN and also GCN with $\mathrm{TopoAug}_{(f_{\mathrm{aux}},\,\mathrm{input})}$. This result suggests that inserting TopoAug's auxiliary features at the output embedding phase is the more appropriate choice. Intuitively, this is because concatenating the auxiliary features with the node features at the input phase can potentially mix up different types of input information, and confuse the GNN.

Table 2: Evaluating different phases for applying TopoAug (TopoAug$_{(f_{\text{aux}},\,\text{input})}$, TopoAug$_{(f_{\text{aux}},\,\text{emb.})}$): accuracy (%) of the TopoAug variants using GCN as the backbone embedding function f_{emb}, whose auxiliary features are integrated at different stages, compared with the vanilla GCN without any augmentation. GCN with TopoAug$_{(f_{\text{aux}},\,\text{emb.})}$ variants show a consistent and much more significant accuracy enhancement compared with vanilla GCN and GCN with TopoAug$_{(f_{\text{aux}},\,\text{input})}$ variants.

Method	GitHub	Brain	Computers
GCN baseline	87.2 ± 0.0	62.5 ± 0.0	75.6 ± 4.1
TopoAug$_{(\text{HyperConv, input})}$	86.9 ± 0.3	63.6 ± 1.3	96.0 ± 0.5
TopoAug$_{(\text{ED-HNN, input})}$	87.1 ± 0.5	63.6 ± 1.3	95.8 ± 0.4
TopoAug$_{(\text{HyperConv, emb.})}$	87.2 ± 0.0	63.7 ± 0.2	96.8 ± 0.5
TopoAug$_{(\text{ED-HNN, emb.})}$	$\mathbf{87.4 \pm 0.3}$	$\mathbf{66.7 \pm 2.6}$	$\mathbf{98.1 \pm 0.7}$

Table 3: Evaluating different types for the auxiliary function f_{aux}: accuracy (%) of TopoAug with various simple-GNNs and hyper-GNNs as auxiliary functions, using GCN as the backbone embedding function f_{emb}, compared with the vanilla GCN without any augmentation. The TopoAug variants with hyper-GNNs as auxiliary functions clearly outperform TopoAug with simple-GNNs.

Method	GitHub	Brain	Computers
GCN baseline	87.2 ± 0.0	62.5 ± 0.0	75.6 ± 4.1
TopoAug$_{(\text{GAT})}$	86.7 ± 0.1	62.5 ± 0.3	91.3 ± 0.1
TopoAug$_{(\text{GraphSAGE})}$	87.0 ± 0.4	65.3 ± 0.9	93.0 ± 0.0
TopoAug$_{(\text{HyperConv})}$	87.2 ± 0.0	63.7 ± 0.2	96.8 ± 0.5
TopoAug$_{(\text{ED-HNN})}$	$\mathbf{87.4 \pm 0.3}$	$\mathbf{66.7 \pm 2.6}$	$\mathbf{98.1 \pm 0.7}$

4.3.2 Choice of the Auxiliary Function Type

It is intuitive to use hyper-GNNs as auxiliary functions (f_{aux}) in TopoAug, so that hyperedge information can be fully captured. However, it is still worth verifying that hyper-GNNs indeed outperform simple-GNNs as auxiliary functions in TopoAug, in order to prove the efficacy of TopoAug. Therefore, we compare the performance of TopoAug with different classes of GNNs as the auxiliary function, denoted as TopoAug$_{(f_{\text{aux}})}$: (1) hyper-GNNs, namely HyperConv and ED-HNN; (2) simple-GNNs, namely GAT [33] and GraphSAGE [13]. These ablation experiments are again conducted on the MUSAE-GitHub, GRAND-Brain and Amazon-Computers datasets, and GCN remains as the GNN for applying TopoAug. *Starting from this stage, all experiments apply TopoAug's auxiliary features at the output embedding phase.* Table 3 reports the results for this set of ablation studies, which clearly show that TopoAug with hyper-GNNs as auxiliary functions outperform TopoAug with simple-GNNs, thereby justifying that TopoAug indeed utilises the virtual hyperedges it constructs.

4.3.3 Identifying the Best Auxiliary Function

Now that the efficacy of TopoAug has been affirmed through the aforementioned ablation studies, we conduct extensive experiments to identify the optimal auxiliary function for TopoAug, denoted as f_{emb}+TopoAug$_{(f_{\text{aux}})}$. We use GCN, GAT and GraphSAGE as the embedding GNNs f_{emb} for applying TopoAug, and test the performance of TopoAug with either HyperConv or ED-HNN as the auxiliary function f_{aux}, on all 23 node classification datasets we build (details of the datasets are described in Section 4.1).

For clarity, we present in Table 4 the average rankings of those six TopoAug applications, together with the three vanilla GNNs without TopoAug. The results clearly show that all TopoAug applications surpass the performance of the vanilla GNNs, which further validate the superior efficacy of TopoAug. Among those TopoAug variants, GraphSAGE with TopoAug$_{(\text{HyperConv})}$ demonstrates to be the most effective combination. In addition, for both GCN and GAT, TopoAug$_{(\text{ED-HNN})}$ outperforms TopoAug$_{(\text{HyperConv})}$, indicating a positive correlation between the expressiveness of the auxiliary hyper-GNN and the performance gain provided by TopoAug. This is likely due to the enhanced ability of more expressive hyper-GNNs to extract hyperedge information, leading to more informative auxiliary features.

Table 4: Evaluating the optimal auxiliary function f_{aux} for TopoAug: average accuracy rankings of three vanilla GNNs and TopoAug with various auxiliary functions on different GNNs, across 23 node classification datasets. GraphSAGE with TopoAug using HyperConv as the auxiliary function proves to be the best-performing model.

Method	Average Ranking
GCN	6.96
GAT	7.91
GraphSAGE	7.09
GCN+TopoAug$_{(\text{HyperConv})}$	5.26
GCN+TopoAug$_{(\text{ED-HNN})}$	3.09
GAT+TopoAug$_{(\text{HyperConv})}$	5.22
GAT+TopoAug$_{(\text{ED-HNN})}$	3.04
GraphSAGE+TopoAug$_{(\text{HyperConv})}$	**2.26**
GraphSAGE+TopoAug$_{(\text{ED-HNN})}$	2.74

4.4 Main Results

We thoroughly evaluate the performance of TopoAug against three simple-GNN baselines: GCN, GAT and GraphSAGE, and two hyper-GNN baselines: HyperConv and ED-HNN, on all 23 node classification datasets we build. We include the two hyper-GNNs as baselines since the augmented combinatorial complexes constructed by TopoAug also make use of the hyperedge information.

Moreover, we also compare the performance of TopoAug with four different existing graph augmentation methods: DropNode [40], DropEdge [28], Mixup [42] and NodeFeatureMasking [9]. We evaluate those graph augmentation methods on seven selected datasets: MUSAE-GitHub and MUSAE-TwitchDE (featuring virtual hyperedge construction from graph statistics); Cora-CoCitation, GRAND-Brain, and GRAND-LungCancer (featuring virtual hyperedge construction from different data perspectives); as well as Amazon-Computers and Amazon-Photos (featuring virtual hyperedge construction from different data modalities). Each of dataset chosen in this comparison is among the largest graphs in their corresponding dataset subgroups. GCN and GraphSAGE are used as the GNNs for the evaluation with the graph augmentation methods. Details about the experimental settings and the full results are provided in [43].

Table 5 summarises the results of the experiments. The results unequivocally demonstrate that *TopoAug consistently outperforms the vanilla GNN baselines and other graph augmentation methods, across all three types of virtual hyperedge construction strategies.* This again highlights the superiority of TopoAug, as well as the idea of incorporating higher-order node relations into graph augmentation methods for real-world complex networks. Besides, TopoAug also outperforms both hyper-GNN baselines, thereby indicating that integrating the original graph with the auxiliary features as augmentations is preferable over only learning from the virtual hyperedges and auxiliary features alone. Among the two backbone embedding GNNs on which TopoAug is applied, GraphSAGE with TopoAug performs better than GCN with TopoAug in general, which is likely due to the more expressive nature of GraphSAGE on those tasks, compared with GCN.

Table 5: Accuracy (%) of TopoAug and corresponding baselines and existing graph augmentation methods on selected node classification datasets. *The results for GCN and GAT on the Cora-CoCitation dataset are directly taken from the GAT paper [33]. **The results for HyperConv and ED-HNN on the Cora-CoCitation dataset are directly taken from the ED-HNN paper [34]. The results clearly show superior performance of TopoAug compared to the baseline GNNs and existing graph augmentation methods.

Method	Graph Statistics		Multi-Perspective			Multi-Modality	
	GitHub	TwitchDE	Cora-CoCitation	Brain	Lung Cancer	Computers	Photos
RandomGuess	*25.0*	*50.0*	*14.3*	*33.3*	*33.3*	*10.0*	*10.0*
GCN	87.2 ± 0.0	65.5 ± 0.2	81.4 ± 0.5*	62.5 ± 0.0	59.6 ± 0.1	75.6 ± 4.1	29.5 ± 1.7
GAT	86.4 ± 0.1	64.5 ± 0.4	83.0 ± 0.7*	62.5 ± 0.1	59.6 ± 0.0	74.2 ± 4.3	43.4 ± 7.4
GraphSAGE	87.1 ± 0.2	65.7 ± 0.1	83.2 ± 0.1	61.8 ± 0.2	61.5 ± 1.5	75.0 ± 1.9	36.6 ± 6.1
HyperConv	80.8 ± 0.1	65.4 ± 0.2	79.1 ± 1.0**	62.5 ± 0.0	59.3 ± 0.3	84.2 ± 2.0	33.7 ± 5.9
ED-HNN	86.2 ± 0.1	68.1 ± 0.6	80.3 ± 1.4**	66.3 ± 1.3	60.2 ± 1.4	97.3 ± 0.2	78.6 ± 1.2
GCN+DropNode	86.2 ± 0.1	67.9 ± 0.7	85.5 ± 1.0	63.3 ± 0.2	58.2 ± 1.3	91.8 ± 2.1	71.1 ± 0.5
GCN+DropEdge	86.6 ± 0.2	67.7 ± 1.0	86.3 ± 0.4	63.2 ± 0.4	60.7 ± 0.3	92.2 ± 0.7	78.6 ± 0.6
GCN+Mixup	85.8 ± 0.3	67.6 ± 0.8	85.6 ± 1.4	65.0 ± 1.0	59.0 ± 0.6	87.7 ± 2.9	78.0 ± 0.1
GCN+NodeFeatureMasking	85.9 ± 0.0	67.8 ± 0.2	85.2 ± 0.9	64.0 ± 1.2	59.2 ± 0.8	91.7 ± 0.3	80.7 ± 0.6
GraphSAGE+DropNode	86.2 ± 0.1	**68.3 ± 0.6**	86.4 ± 0.0	63.3 ± 0.1	64.5 ± 0.8	95.9 ± 0.2	69.8 ± 0.5
GraphSAGE+DropEdge	86.8 ± 0.1	67.8 ± 1.3	87.1 ± 0.1	64.2 ± 0.2	64.7 ± 0.7	95.9 ± 0.3	80.5 ± 0.6
GraphSAGE+Mixup	85.9 ± 0.3	67.1 ± 0.2	**87.2 ± 1.3**	64.2 ± 1.5	63.4 ± 0.1	92.2 ± 0.0	71.5 ± 0.3
GraphSAGE+NodeFeatureMasking	86.1 ± 0.0	68.1 ± 0.7	87.1 ± 0.5	64.4 ± 0.4	64.9 ± 2.3	95.7 ± 0.2	79.3 ± 0.8
GCN+TopoAug	**87.4 ± 0.3**	67.9 ± 0.9	86.6 ± 1.4	**66.7 ± 2.6**	63.7 ± 0.9	98.1 ± 0.7	**80.9 ± 0.3**
GraphSAGE+TopoAug	87.3 ± 0.2	**68.3 ± 0.3**	**87.2 ± 1.8**	66.6 ± 1.6	**66.4 ± 0.2**	**98.2 ± 0.7**	**80.9 ± 0.7**

Table 6: MSE (↓) of TopoAug and three vanilla GNNs on the node regression datasets (MUSAE-Wiki). The results clearly show superior performacne of TopoAug compared to the vanilla GNN baselines.

Method	Squirrel	Crocodile	Chameleon
GCN	7.319 ± 0.000	8.761 ± 0.001	6.779 ± 0.005
GAT	7.313 ± 0.007	8.093 ± 0.054	6.249 ± 0.261
HyperConv	7.230 ± 0.002	8.706 ± 0.000	6.712 ± 0.001
GCN+TopoAug	6.557 ± 0.154	**4.851 ± 0.014**	5.515 ± 0.008
GAT+TopoAug	**6.049 ± 0.204**	4.875 ± 0.031	**4.665 ± 0.050**

It is noted that on the MUSAE-TwitchDE and Cora-CoCitation datasets, although TopoAug is still the best performing method, its performance improvements compared to other graph augmentation methods are not as significant as on other datasets. This can be attributed to the following specific characteristics of the two datasets:

- The MUSAE-TwitchDE and Cora-CoCitation datasets are comparatively small containing a few thousands of nodes, and most appropriate graph augmentation methods can help the GNNs saturate in extracting their information.
- The virtual hyperedges for the MUSAE-TwitchDE dataset are constructed by computing from the original graph's statistics, which does not introduce new information to the graph, thus also limiting TopoAug's effectiveness in leveraging GNNs' performance.
- The virtual hyperedges for the Cora-CoCitation dataset are constructed by grouping papers with co-authors, which are not diverse to the co-citation links and may contain redundant information, compared with the GRAND (gene regulations vs. geometrical information) and Amazon (text vs. image modality) datasets.

These findings provide valuable guidance for employing TopoAug in real-world applications, in order to fully unleash its potential.

In addition, we also perform supplementary evaluation of TopoAug on the three node regression datasets: MUSAE-Wiki-Chameleon, MUSAE-Wiki-Crocodile and MUSAE-Wiki-Squirrel, against the GCN, GAT and HyperConv baselines, and report their MSEs in Table 6. The results also show consistently and significantly lower MSEs when TopoAug is applied, which further validate the superiority of TopoAug. On those node regression datasets, GAT with TopoAug generally outperforms GCN with TopoAug, since GAT is more expressive than GCN on those tasks. This observation also aligns well with the results on the node classification datasets, which suggests that TopoAug is stable across various types of tasks.

5 Conclusion

This work introduces *Topological Augmentation (TopoAug)*, a novel graph augmentation method designed to improve the performance of GNNs in node prediction tasks across diverse real-world large-scale graph datasets. TopoAug enhances the capability of GNNs by constructing combinatorial complexes from the original graphs using virtual hyperedges, and then generating auxiliary features for each node. This improvement allows even basic GNN models to surpass the limitations of the 1-WL test. Empirical results confirm that TopoAug outperforms traditional graph augmentation methods in terms of node prediction across various data domains.

For future work, there are several potential methods that could be explored to further utilise the auxiliary information. These include applying a cross-attention mechanism, as detailed in works [32, 36], to effectively integrate the auxiliary and original features. Another unexplored avenue is the concurrent optimisation of the auxiliary and original features using contrastive loss, a technique discussed in ViLT [18]. Employing these methods has the potential to optimise the integration and utility of the auxiliary features within GNNs, thereby further increasing their overall performance.

Acknowledgements

This work was performed using the Sulis Tier 2 HPC platform hosted by the Scientific Computing Research Technology Platform at the University of Warwick, the JADE Tier 2 HPC facility, and the Cirrus UK National Tier-2 HPC Service at EPCC. Sulis is funded by EPSRC Grant EP/T022108/1 and the HPC Midlands+ consortium. JADE is funded by EPSRC Grant EP/T022205/1. Cirrus is funded by the University of Edinburgh and EPSRC Grant EP/P020267/1. Xiangyu Zhao acknowledges the funding from the Imperial College

London Electrical and Electronic Engineering PhD Scholarship. Zehui Li acknowledges the funding from the UKRI 21EBTA: EB-AI Consortium for Bioengineered Cells & Systems (AI-4-EB) award, Grant BB/W013770/1. Mingzhu Shen acknowledges the funding from the Imperial College London President's PhD Scholarship. Furthermore, the authors would also like to thank Jing Zeng for her careful proofreading of this manuscript.

References

[1] S. Bai, F. Zhang, and P. H. Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021.

[2] M. Ben Guebila, C. M. Lopes-Ramos, D. Weighill, A. R. Sonawane, R. Burkholz, B. Shamsaei, J. Platig, K. Glass, M. L. Kuijjer, and J. Quackenbush. GRAND: a database of gene regulatory network models across human conditions. *Nucleic Acids Research*, 50(D1):D610–D621, 2022.

[3] S. Brody, U. Alon, and E. Yahav. How attentive are graph attention networks? In *The 10th International Conference on Learning Representations*, 2022.

[4] C. Cai and Y. Wang. A note on over-smoothing for graph neural networks. *ICML 2020 Graph Representation Learning and Beyond Workshop*, 2020.

[5] T.-H. H. Chan and Z. Liang. Generalizing the hypergraph laplacian via a diffusion process with mediators. *Theoretical Computer Science*, 806: 416–428, 2020.

[6] K. Ding, Z. Xu, H. Tong, and H. Liu. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, 24 (2):61–77, 2022.

[7] Y. Dong, W. Sawin, and Y. Bengio. HNHN: Hypergraph networks with hyperedge neurons. *ICML 2020 Graph Representation Learning and Beyond Workshop*, 2020.

[8] G. Eraslan, Ž. Avsec, J. Gagneur, and F. J. Theis. Deep learning: new computational modelling techniques for genomics. *Nature Reviews Genetics*, 20(7):389–403, 2019.

[9] F. Feng, X. He, J. Tang, and T.-S. Chua. Graph adversarial training: Dynamically regularizing based on graph structure. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2493–2504, 2019.

[10] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao. Hypergraph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3558–3565, 2019.

[11] A. Gotmare, N. S. Keskar, C. Xiong, and R. Socher. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. In *The 7th International Conference on Learning Representations*, 2019.

[12] M. Hajij, G. Zamzmi, T. Papamarkou, N. Miolane, A. Guzmán-Sáenz, K. N. Ramamurthy, T. Birdal, T. K. Dey, S. Mukherjee, S. N. Samaga, et al. Topological deep learning: Going beyond graph data. *arXiv preprint arXiv:2206.00606*, 2022.

[13] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 30, pages 1024–1034. Curran Associates, Inc., 2017.

[14] W. L. Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159, 2020.

[15] X. Han, Z. Jiang, N. Liu, and X. Hu. G-mixup: Graph data augmentation for graph classification. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 8230–8248. PMLR, 2022.

[16] R. He and J. McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, pages 507–517. International World Wide Web Conferences Steering Committee, 2016.

[17] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems*, volume 33, pages 22118–22133. Curran Associates, Inc., 2020.

[18] W. Kim, B. Son, and I. Kim. ViLT: Vision-and-language transformer without convolution or region supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 5583–5594, Virtual, 2021. PMLR.

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *The 3rd International Conference on Learning Representations, Conference Track Proceedings*, 2015.

[20] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *The 5th International Conference on Learning Representations, Conference Track Proceedings*, 2017.

[21] G. Liu, E. Inae, T. Zhao, J. Xu, T. Luo, and M. Jiang. Data-centric learning from unlabeled graphs with diffusion model. In *Advances in Neural Information Processing Systems*, pages 21039–21057. Curran Associates, Inc., 2023.

[22] Y. Long, X. Wang, D. T. Youmans, and T. R. Cech. How do lncRNAs regulate transcription? *Science Advances*, 3(9):eaao2110, 2017.

[23] G. A. Maston, S. K. Evans, and M. R. Green. Transcriptional regulatory elements in the human genome. *Annual Review of Genomics and Human Genetics*, 7:29–59, 2006.

[24] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52. ACM, 2015.

[25] F. Monti, F. Frasca, D. Eynard, D. Mannion, and M. M. Bronstein. Fake news detection on social media using geometric deep learning. *ICLR 2019 Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[26] J. Ni, J. Li, and J. McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 188–197. ACL, 2019.

[27] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 8748–8763, Virtual, 2021. PMLR.

[28] Y. Rong, W. Huang, T. Xu, and J. Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *The 8th International Conference on Learning Representations*, 2020.

[29] B. Rozemberczki, C. Allen, and R. Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

[30] T. K. Rusch, M. M. Bronstein, and S. Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023.

[31] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann. Pitfalls of graph neural network evaluation. *NeurIPS 2018 Relational Representation Learning Workshop*, 2018.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc., 2017.

[33] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *The 6th International Conference on Learning Representations, Conference Track Proceedings*, 2018.

[34] P. Wang, S. Yang, Y. Liu, Z. Wang, and P. Li. Equivariant hypergraph diffusion neural operators. In *The 11th International Conference on Learning Representations*, 2023.

[35] Y. Wang, W. Wang, Y. Liang, Y. Cai, and B. Hooi. Mixup for node and graph classification. In *Proceedings of the Web Conference 2021*, pages 3663–3674. ACM, 2021.

[36] X. Wei, T. Zhang, Y. Li, Y. Zhang, and F. Wu. Multi-modality cross attention network for image and sentence matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10941–10950. IEEE, 2020.

[37] B. Weisfeiler and A. Leman. A reduction of a graph to canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia*, 2(9):12–16, 1968.

[38] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *The 7th International Conference on Learning Representations*, 2019.

[39] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar. HyperGCN: A new method for training graph convolutional networks on hypergraphs. In *Advances in Neural Information Processing Systems*, volume 32, pages 1511–1522. Curran Associates, Inc., 2019.

[40] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen. Graph contrastive learning with augmentations. In *Advances in Neural Information Processing Systems*, volume 33, pages 5812–5823. Curran Associates, Inc., 2020.

[41] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna. GraphSAINT: Graph sampling based inductive learning method. In *The 8th International Conference on Learning Representations*, 2020.

[42] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *The 6th International Conference on Learning Representations, Conference Track Proceedings*, 2018.

[43] X. Zhao, Z. Li, M. Shen, G.-B. Stan, P. Liò, and Y. Zhao. Enhancing node representations for real-world complex networks with topological augmentation. *arXiv preprint arXiv:2402.13033*, 2024.