ECAI 2023
K. Gal et al. (Eds.)

© 2023 The Authors.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/FAIA230325

Data-Driven Self-Supervised Graph Representation Learning

Ahmed E. Samy[a;*], **Zekarias T. Kefato**[a] **and Šarūnas Girdzijauskas**[a]

[a]KTH, Royal Institute of Technology, Stockholm, Sweden
aesy@kth.se, zekarias@kth.se, sarunasg@kth.se

Abstract. Self-supervised graph representation learning (SSGRL) is a representation learning paradigm used to reduce or avoid manual labeling. An essential part of SSGRL is graph data augmentation. Existing methods usually rely on heuristics commonly identified through trial and error and are effective only within some application domains. Also, it is not clear why one heuristic is better than another. Moreover, recent studies have argued against some techniques (e.g., *dropout*: that can change the properties of molecular graphs or destroy relevant signals for graph-based document classification tasks).

In this study, we propose a novel data-driven SSGRL approach that automatically learns a suitable graph augmentation from the signal encoded in the graph (i.e., the nodes' predictive feature and topological information). We propose two complementary approaches that produce learnable feature and topological augmentations. The former learns multi-view augmentation of node features, and the latter learns a high-order view of the topology. Moreover, the augmentations are jointly learned with the representation. Our approach is general that it can be applied to homogeneous and heterogeneous graphs. We perform extensive experiments on node classification (using nine homogeneous and heterogeneous datasets) and graph property prediction (using another eight datasets). The results show that the proposed method matches or outperforms the SOTA SSGRL baselines and performs similarly to semi-supervised methods. The anonymised source code is available at https://github.com/AhmedESamy/dsgrl/

1 Introduction

Self-supervised graph representation learning (SSGRL) has been successfully used for graph representation learning (GRL) [12, 17, 26, 29, 37] in various domains where labeled data is scarce and manual label is expensive. It has recently attracted interest across domains by achieving a competitive performance when compared to semi-supervised approaches. Considering the scarcity of labeled data, SSGRL has emerged as a new paradigm that narrows down the performance gap between the unsupervised and semi-supervised learning methods.

Self-supervised learning (SSL) is commonly formulated as a *predictive* or *contrastive* learning [46]. For predictive models [7], one first defines a related task on which an SSL model is pre-trained to extract meaningful patterns. The pre-trained model is subsequently refined (fine-tuned) on a relevant but specific task of interest. Typically, an SSL model is pre-trained over large data as a starting point. The quintessential models, particularly from NLP, are the ones that are pre-trained on masked word prediction tasks and are fine-tuned on other relevant tasks, such as text classification or translation.

On the other hand, contrastive models learn based on augmented views of a data point (e.g., image, graph) that are generated by applying a *meaningful* perturbation on the original data point. The representation of a data point is then learned by maximizing the mutual information between latent representations obtained from its augmented views. The main challenge here is to produce augmented views of the data points.

The key to learning high-quality representations based on augmentation is that the perturbations should preserve semantics [1, 6, 38, 49]. For instance, a perturbation applied to an image of a dog should preserve *"dogness"*. Effective augmentation techniques for images (e.g., rotation, flipping, resizing) allow learning high-quality visual representations because they preserve the semantics of the original image. This is also true for SSL techniques in NLP [7,21,23], (e.g. synonym augmentation and word masking), such techniques do not alter the meaning of the original sentence.

Due to the complex nature of graph data, it is much more challenging to find appropriate techniques for augmenting graphs. While some techniques are proposed, there is no standard technique that works well for graphs in different domains [14, 18, 35, 38–40, 48, 53]. Consequently, most efforts rely on finding a heuristic by trial and error to identify a suitable augmentation for the graph at hand.

Generally, there are two classes of perturbations, which either corrupt the topology of the graph or node features. The topology can be corrupted by dropping nodes and edges or adding new edges either randomly or through a diffusion process [14, 35, 40, 53]. Similarly, dropout, masking, and permutation techniques have been used for corrupting node features [18, 35]. Nonetheless, it is unclear why a particular augmentation technique works better. A study [39] has shown that these strategies are susceptible to destroying task-relevant information. Furthermore, in some cases, e.g., for molecular graphs, dropout techniques alter the semantics of the graph [46]. The effectiveness of such techniques usually comes not from the graph augmentations but from the strong inductive bias of the underlying learning algorithm, particularly Graph Neural Networks (GNNs) [39].

In this paper, we follow a data-driven approach, where the augmentation process is guided by the inherent signal encoded in the graph. Such an approach establishes obvious benefits, first as one can avoid trial and error in identifying a suitable augmentation mechanism. Second, it provides a flexible framework that can be adapted to different domains.

Thus, we propose a novel **D**ata-driven **S**elf-supervised **G**raph **R**epresentation **L**earning (DSGRL) method. DSGRL is data-driven because it jointly learns the augmentation with the representation. Similar to existing methods, we aim to augment either the topology or node features; however, unlike them DSGRL learns both augmentations from the data. DSGRLis a *general* approach that can be applied to both homogeneous and heterogeneous graphs (i.e., graphs containing multiple node/edge types).

Generally, for a given graph G, and a family of augmentation heuristics $\mathcal{A}$, existing methods apply either a topological, $A_t \sim \mathcal{A}$, or feature, $A_f \sim \mathcal{A}$, augmentation sampled from $\mathcal{A}$. However, DSGRL does not rely on $\mathcal{A}$,

* Corresponding Author. Email: aesy@kth.se

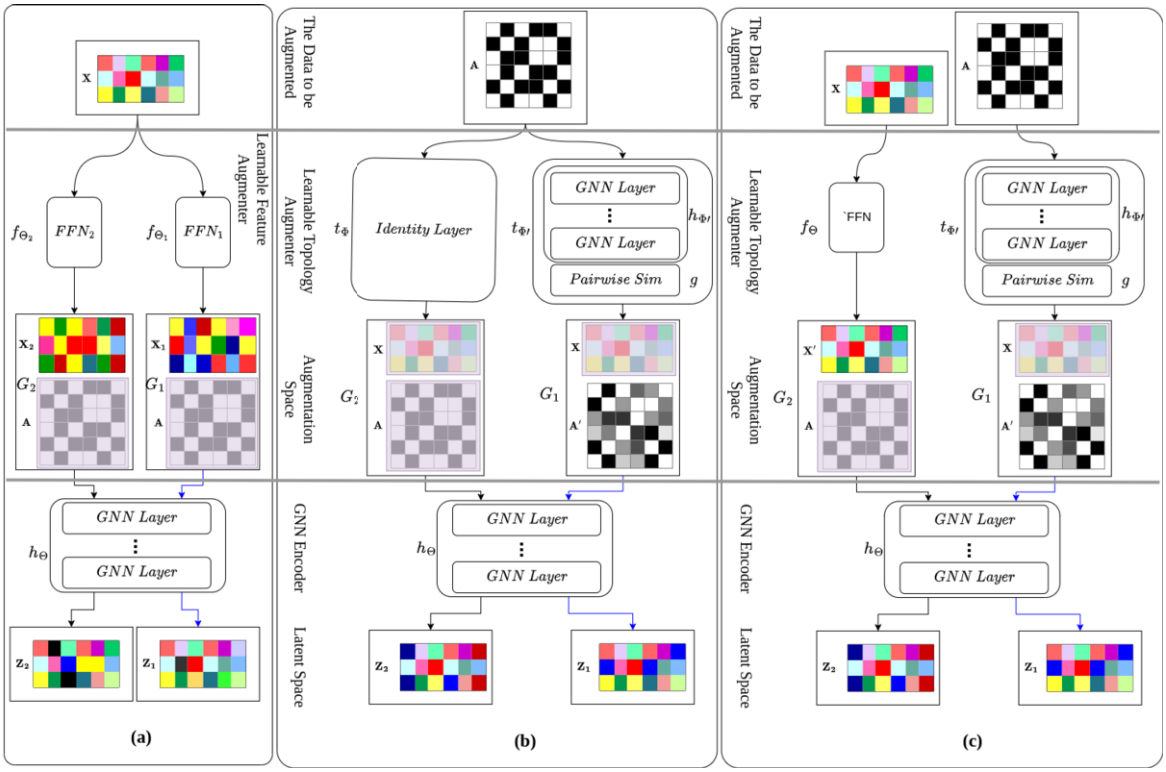

Figure 1. The Architecture of DSGRL . The left figure (a) is based on learnable feature augmentation, where we only augment node features, $\mathbf{X}$. Two learnable augmenters $f_{\Theta_1}(\mathbf{X})$ and $f_{\Theta_2}(\mathbf{X})$ are applied on $\mathbf{X}$ to obtain the learned augmentations $\mathbf{X}_1$ and $\mathbf{X}_2$, respectively. The middle figure (b) is used for learning topological augmentation. Two augmentations, an identical, $t_\Phi(\mathbf{A})$, and learned ones, $t_{\Phi'}(\mathbf{A})$, are respectively applied on the adjacency matrix to obtain $\mathbf{A}$ and a high-order network $\mathbf{A}'$. The right figure (c) is used for combining learnable feature and topological augmentations. The augmentation results in two views $G_1=(\mathbf{A}_1,\mathbf{X}_1)$ and $G_2=(\mathbf{A}_2,\mathbf{X}_2)$ from the augmentation space, where $\mathbf{A}_1,\mathbf{X}_1,\mathbf{A}_2,\mathbf{X}_2$ are set to different values as shown in the above figures based on the augmentation type. Finally, a shared GNN encoder h_Θ is applied on the views, $h_\Theta(\mathbf{A}_1,\mathbf{X}_1)$ and $h_\Theta(\mathbf{A}_2,\mathbf{X}_2)$, respectively, to obtain the latent representations $\mathbf{Z}_1$ and $\mathbf{Z}_2$. The data modality ($\mathbf{A}$ or $\mathbf{X}$) that is not affected by an augmenter is blurred.

instead it learns A_{t_Φ} and A_{f_Θ}; where Φ and Θ are the learnable parameters of the topological and feature augmenters, respectively. We materialize t_Φ using a GNN to obtain high-order node features; based on similarity scores between these features, we compute a high-order weighted network as a topological augmentation. The feature augmentation, f_Θ, is simply a feed-forward neural network (FFN). Note that, A_{f_Θ} in Fig. 1(a) and A_{t_Φ} in Fig. 1(b) are technically complementary and *not competitive*. One can combine both augmentations at the same time as shown in Fig. 1(c). Next, given two augmented views G_1 and G_2 of a graph G, the latent representation $\mathbf{Z}=AGG(\mathbf{Z_1},\mathbf{Z_2})$ of the graph is obtained by applying a shared GNN – h_θ, $\mathbf{Z}_1=h_\theta(G_1)$ and $\mathbf{Z}_2=h_\theta(G_2)$. G_1 and G_2 are learned using either topological or feature augmentation or a combination thereof.

Contrastive models require a negative (contrastive) term to prevent collapse, and negative sampling is the most common strategy for this [4,14,15,33–35,40,43,47,52,53]. However, it has two limitations; first, it requires a large batch size, and second, sampling truly negative terms is difficult. Alternatives that do not require explicit negative sampling have been proposed to overcome such limitations [6, 18, 38]. Nevertheless, the alternatives are usually based on engineering tricks. In this study, we closely follow a recent method that uses a principled approach based on variance, invariance and covariance to prevent collapse [1].

We perform extensive experiments using nine node classification datasets, including homogeneous and heterogeneous datasets, and another eight graph property predictions. We compare our method against seven popular SOTA SSGRL methods. The results show that DSGRL matches or improves the SSGRL baselines, and it is comparable with the semi-supervised methods.

2 The Proposed Method

2.1 *Preliminary*

We consider a graph $G=(\mathbf{A},\mathbf{X})$ with a set of N nodes V and M edges E. $\mathbf{A}\in\{0,1\}^{N\times N}$ denotes the adjacency matrix of G and $\mathbf{X}\in\mathbb{R}^{N\times F}$ is a feature matrix, where F is the number of features. For a given row index i, $\mathbf{A}[i]=\mathbf{a}_i$ and $\mathbf{X}[i]=\mathbf{x}_i$ represent the topological structure and feature signals of node i. For any index i, $\mathbf{A}[:,i]=\mathbf{a}_{:i}$ and $\mathbf{X}[:,i]=\mathbf{x}_{:i}$ refers to the indexing of the i^{th} column of the adjacency and feature matrices, respectively. Finally, $\mathbf{z}_{ij}$ corresponds to the ij^{th} entry of any matrix $\mathbf{Z}$.

We consider a message passing GNN, h_Θ, is given, and

$$h_\Theta(G)=h_\Theta(\mathbf{A},\mathbf{X})=\sigma(...\sigma(\mathbf{A}'\mathbf{X}^{(l)}\mathbf{W}^{(l)})...)$$

where, $\mathbf{W}^{(l)}\in\Theta$ is the weight matrix of the l^{th} layer, σ is an activation function, e.g., `ReLU`, and $\mathbf{A}'$ is a transformed adjacency matrix. Depending on the type of GNN, one can apply different transformations on $\mathbf{A}$, e.g., the symmetric normalization used in [19].

2.2 *The Case for* DSGRL

Several techniques for self-supervised graph representation learning (SSGRL) rely on perturbations by randomly dropping nodes, edges, or subgraphs. This perturbation is acceptable for social graphs. However, they

are susceptible to losing semantics. For instance, dropping a node (an atom) or an edge (a bond) from a particular molecule could alter the essential properties of the molecule [46]. Furthermore, a recent study [39] shows that even for other tasks, e.g., document representation learning based on graphs, dropout techniques could destroy task-relevant information.

We propose graph data augmentation techniques that are inspired by recent studies that advocate for learning augmentation governed by the graph signal or context.

2.3 Learnable Augmentation

The key hypothesis behind learning augmentations is that because it is a data-driven approach, it enables us to effectively capture augmentation signals without the human intervention needed for heuristics based on trial and error. Therefore, we propose two alternative approaches, which are learnable *feature* and *topology* augmentation.

2.3.1 Learnable Feature Augmentation

This technique allows us to learn node feature augmentations. Given a graph $G=(\mathbf{A},\mathbf{X})$, we apply a learnable feature augmentation A_{f_Θ} on G as:

$$A_{f_\Theta}(G)=(\mathbf{A},f_\Theta(\mathbf{X}))$$

and we model f as a feed-forward neural network (FFN). Two separate learnable functions f_{Θ_1} and f_{Θ_2}, parameterized by Θ_1 and Θ_2 compute two augmented views of $\mathbf{X}$ as:

$$\mathbf{X_1}=f_{\Theta_1}(\mathbf{X})\in\mathbb{R}^{B\times D_1}$$
$$\mathbf{X_2}=f_{\Theta_2}(\mathbf{X})\in\mathbb{R}^{B\times D_1}$$

where

$$\Theta_1=\{\mathbf{W}_1^{(l)}:l=1,...,L\},\Theta_2=\{\mathbf{W}_2^{(l)}:l=1,...,L\}$$

are set of weights, and $\mathbf{W}_1^{(l)}$ or $\mathbf{W}_2^{(l)}$ are the weight matrices of the l–th layer of the FFNs, B is batch-size, D_1 is the augmentation dimension, and L is the number of layers of the FFNs. Figure 1 (a) shows DSGRL's architecture based on learnable feature augmentation.

2.3.2 Learnable Topology Augmentation

Studies have shown that using diffusion-based high-order networks improves the performance of GNNs [20]. Consequently, high-order networks have been used for augmentation in SSGRL. This study proposes a complementary approach that learns the K–order relation between nodes. That is, we apply $A_{t'_\Phi}$ on G as:

$$A_{t'_\Phi}(G)=(t'_\Phi(\mathbf{A}),\mathbf{X})$$

to obtain a high-order network

$$\mathbf{A}'=t'_\Phi(\mathbf{A})$$

First, we learn a latent representation, $\mathbf{H}\in\mathbb{R}^{B\times D_1}$, which encodes high-order (K–hop) signal. To this end, we employ a GNN, as GNNs enable us to receive a signal from K–hop neighbors similar to static diffusion algorithms, such as personalized PageRank and heat kernel. Hence, for each node i, a GNN h_Φ, parameterized by Φ is used to learn high-order feature vector $\mathbf{H}[i]=\mathbf{h}_i$ and $\mathbf{H}$ is computed as:

$$\mathbf{H}=h_\Phi(\mathbf{A},\mathbf{X})$$

where $\Phi=\{\mathbf{W}^{(l)}:l=1,...,K\}$, $\mathbf{W}^{(l)}$ is the weight matrix of the l-th layer of the GNN, and K is the number of layers.

We obtain two topological views, which are $\mathbf{A}'=t_{\Phi'}(\mathbf{A})$ and $\mathbf{A}=t_\Phi(\mathbf{A})$, where t_Φ is simply an identity augmentor. The high-order network $\mathbf{A}'$ is constructed based on the high-order features as

$$\mathbf{A}'=t'_\Phi(\mathbf{A})=g(\mathbf{H},\mathbf{H})$$

The entry $\mathbf{a}'_{ij}$ is computed as

$$\mathbf{a}'_{ij}=\begin{cases}g(\mathbf{h}_i,\mathbf{h}_j), & \text{if } g(\mathbf{h}_i,\mathbf{h}_j)>\mathbb{E}_{k\in V}[g(\mathbf{h}_i,\mathbf{h}_k)]\\ 0, & \text{otherwise}\end{cases}$$

and g is defined as

$$g(\mathbf{h}_i,\mathbf{h}_j)=\mathbf{h}_i^T\cdot\mathbf{h}_j$$

DSGRL's architecture based on this augmentation technique is depicted in Fig. 1(b).

2.4 Encoding

After generating two views of the graph, either using the feature or topology augmentor, we feed each view independently to a shared GNN encoder, h_Θ, to learn a latent graph representation. For brevity, regardless of the augmentor, we refer to the views in the augmentation space as $G_1=(\mathbf{A}_1,\mathbf{X}_1)$ and $G_2=(\mathbf{A}',\mathbf{X}_2)$. The next task is to learn two latent representations $\mathbf{Z_1}\in\mathbb{R}^{B\times D}$ and $\mathbf{Z_2}\in\mathbb{R}^{B\times D}$ that encode the two views, where D is the number of latent dimensions. We achieve this by using a shared GNN, h_Θ, as:

$$\mathbf{Z_1}=h_\Theta(\mathbf{A},\mathbf{X}_1)$$
$$\mathbf{Z_2}=h_\Theta(\mathbf{A}',\mathbf{X}_2)$$

For full-batch training, the batch axis becomes N instead of B. Henceforth though, we assume a mini-batch training.

2.5 Training

Generally, in SSGRL, we want the latent representations $\mathbf{Z}_1$ and $\mathbf{Z}_2$ of nodes to be invariant to the perturbations. For this reason, we want to maximize the agreement (similarity) between $\mathbf{Z}_1$ and $\mathbf{Z}_2$. Minimizing the L-2 distance is commonly used for this purpose; thus, we use the same strategy. We closely follow a similar formulation as [1] and define a term called invariance based on the L-2 distance as:

$$\texttt{inv}=||\mathbf{Z}_1-\mathbf{Z}_2||_F \tag{1}$$

Nonetheless, this has a trivial solution that collapses the representations. Several strategies, mostly engineering tricks, have been used to prevent this collapse [11, 18, 38]. Instead, we use a principled approach inspired by a recent method [1] proposed for visual representation. That is, we add two regularization terms called variance and covariance regularizations.

The variance regularization is defined as

$$v(\mathbf{Z})=\frac{1}{D}\sum_{j=1}^{D}\max(0,1-\sqrt{Var(\mathbf{z}_{:j})+\epsilon}) \tag{2}$$

and it constrains each dimension of the latent representation to have a variance of 1; as a result prevents data points from collapsing into a subspace.

The covariance term is defined as

$$c(\mathbf{Z})=\frac{1}{D}\sum_{i\neq j}\Big[\frac{\bar{\mathbf{Z}}^T\bar{\mathbf{Z}}}{B-1}\Big]_{i,j}^2 \tag{3}$$

and it is the sum of the squared-off diagonal elements of the covariance matrix $\bar{\mathbf{Z}}^T\bar{\mathbf{Z}}$, where $\bar{\mathbf{Z}}$ is the mean centered representation. The covariance term is normalized and scaled with respect to B and D, respectively. As a result of constraining the off-diagonal elements of the covariance matrix to be zero, this regularization ensures that each dimension is independent of each other, consequently preventing the dimensions from collapsing. Finally, we define the regularization on the latent space as:

$$R_{\mathbf{Z}_1,\mathbf{Z}_2} = \beta * (v(\mathbf{Z}_1) + v(\mathbf{Z}_2)) + \gamma * (c(\mathbf{Z}_1) + c(\mathbf{Z}_2)) \tag{4}$$

Furthermore, the augmentation models can collapse, that is, $f_{\Theta_1} = f_{\Theta_2}$; albeit, empirically, this has not been observed. Since the regularizations mentioned above on the latent space only ensure that neither $\mathbf{Z}_1$ nor $\mathbf{Z}_2$ collapse along the batch or the dimension axes. Thus, to prevent model collapse, we define a model regularization term as:

$$R_{\Theta_1,\Theta_2} = \sum_{\mathbf{W}_l} ||\mathbf{W}_l\mathbf{W}_l^T - \mathbf{I}||_F \tag{5}$$

where $\mathbf{W}_l = \begin{bmatrix}\mathbf{W}_1^{(l)} \\ \mathbf{W}_2^{(l)}\end{bmatrix}$ is a vertical stacking of $\mathbf{W}_1^{(l)} \in \Theta_1$ and $\mathbf{W}_2^{(l)} \in \Theta_2$; recall that $\mathbf{W}_1^{(l)}$ and $\mathbf{W}_2^{(l)}$ are weights of the l^{th} layer of a FFN. For any two row indices i and j of $\mathbf{W}_l$, where $i \neq j$, the model regularization encourages each row vector $\mathbf{W}_l[i]$ to be orthogonal to any other vector $\mathbf{W}_l[j]$. Consequently, $\nexists i : \mathbf{W}_1^{(l)}[i] = \mathbf{W}_2^{(l)}[i] \Rightarrow \mathbf{W}_1^{(l)} \neq \mathbf{W}_2^{(l)}$.

The overall training cost function is then defined as:

$$\mathcal{L}_\Psi = \alpha * \texttt{inv} + R_{\mathbf{Z}_1,\mathbf{Z}_2} + \lambda * R_{\Theta_1,\Theta_2} \tag{6}$$

where Ψ is the set of all model parameters, that is, $\Psi = \{\Theta, \Theta_1, \Theta_2\}$ for the model based on feature augmentation and $\Psi = \{\Theta, \Phi\}$ for the model based on topology augmentation. The coefficients α, β, γ, and λ control the contribution of the different cost function terms. In most cases, we have observed that setting these values to just one works well.

If one desires to reduce the number of hyper-parameters, we provide an alternative formulation for Eq. 4. By following the same formulation as Eq. 5, an alternative formulation that is inspired by Laplacian Eigenmaps [2] is defined:

$$R_{\mathbf{Z}_1,\mathbf{Z}_2} = \gamma * (||\tilde{\mathbf{Z}}_1\tilde{\mathbf{Z}}_1^T - \mathbf{I}||_F + ||\tilde{\mathbf{Z}}_2\tilde{\mathbf{Z}}_2^T - \mathbf{I}||_F) \tag{7}$$

where $\mathbf{I} \in \mathbf{R}^{B \times B}$ is an identity matrix and each row vector $\tilde{\mathbf{z}}_\mathbf{i}$ of $\tilde{\mathbf{Z}}$ is a unit vector. We refer to Eq. 7 as an orthogonality regularization, and empirically it performs similarly to Eq. 4. The main disadvantage of Eq. 7 is that it could be expensive for full-batch GNNs, where $B = N$.

Note that DSGRL is jointly optimized on both the augmenter and encoder parameters. As a result, the learned augmentations are governed by the inherent signal in the data.

3 Empirical Evaluation

We validate the proposed method on node classification (NC) and graph property prediction (GPP) tasks. In the former case, the prediction is at a node level, and for the latter, it is at a graph level. Additional thorough analysis of the experiments and the running times of the methods and are included in the appendix. [1]

3.1 Datasets

The datasets are 8 for NC and 8 for GPP, and a summary is provided in Tables 1 and 2.

[1] https://github.com/AhmedESamy/dsgrl/blob/main/Appendix.pdf

3.1.1 NC Datasets

- Citation Networks (PubMed): Paper-to-paper citation networks, and we classify papers into different subjects [13].
- Co-Author Networks (MAG-CS): Author collaboration network from Microsoft Academic Graph, and the task is to predict the active field of authors [31].
- Co-Purchased Products Network (AmazonPhoto): Co-purchased products from Amazon Photo Category, and the task is to predict the refined categories [31].
- Wikipedia (WikiCS): Wikipedia hyperlinks between Computer Science articles, and we classify articles into branches of CS [31].
- Social (Facebook, GitHub, Reddit, and Yelp): Facebook contains a page-to-page graph of verified Facebook sites, and we want to classify pages into their categories [28]. GitHub contains the social network of developers, and we want to classify developers as web or machine learning developers [28]. Yelp is also the social network of Yelp users, and we predict the business categories each user has reviewed. For Reddit, we predict the subreddits (communities) of user posts [13, 50].

3.1.2 GPP Datasets

- Chemical Datasets (DD, NCI1, PROTEINS, ENZYMES) [24] that represent protein interaction or molecular graphs. The task is to predict different properties of molecules or macromolecules.
- Social Datasets (IMDB-BINARY, IMDB-MULTI, REDDIT-BINARY, COLLAB) [24] that represent the collaboration between users in different ego-networks. The task is to predict the class of the ego-networks.

3.2 Baselines

We compare our method against strong self-supervised baselines. As a result of a plethora of related methods, baselines are selected either based on their popularity or if they are current SOTA methods that outperform existing methods. Hence, for the NC task, we select DGI [40], a method that uses corruption based on permutation of node features and topology, MVGRL [14], which augments the topology using high-order networks obtained through a diffusion process, and GCA [53] is a method based on adaptive edge removal and feature masking augmentations. All of these use a contrastive architecture using mutual information maximization with negative sampling. Furthermore, we include a contrastive architecture based on asymmetry called BGRL [11] for completeness. Although it was initially proposed for visual representations, recent studies [18,38] have extended it for GRL. We use the best augmentations reported in these papers.

As there are several studies for the GPP task, we select strong representative baselines, which are GRAPHCL [47, 48], ADGCL [35], and INFOGRAPH [33]. For example, SimGrace [45] is shown to give sub-optimal performance compared to GraphCL [47] as reported in [22, 36], therefore, we only include the results for GraphCL in our experiments. GRAPHCL learns augmentations from a set of augmentations whereas ADGCL learns to drop edges using adversarial training. INFOGRAPH learns by maximizing the mutual information between graph and patch (subgraphs, node, edges) level representations.

Furthermore, we include untrained variants of DSGRL as suggested in [39]. We refer to them as Random-F and Random-T to denote the feature and topology augmentation-based architectures, respectively.

All the above baselines are self-supervised and unsupervised methods. We also include semi-supervised methods; however, they are included just for reference and not comparison.

Table 1. Summary of the datasets used for node classification experiment. BCC, MCC, and MLC refer to the classification task, which is binary, multi-class, and multilabel classification, respectively

Dataset	MAG-CS	AmazonPhoto	PubMed	GitHub	WikiCS	Dezer	Yelp	Reddit
N	18,333	7,650	19,717	37,700	11,701	28,281	716,847	232,965
M	163,788	238,162	88,648	578,006	297,110	185,504	13,954,819	114,615,892
F	6,805	745	500	128	300	128	300	602
#classes	15 (MCC)	8 (MCC)	3 (MCC)	2 (BCC)	15 (MCC)	2 (BCC)	100 (MLC)	41 (MCC)

Table 2. Summary of the dataset used for graph property prediction task, #G is the number of graphs and $\tilde{N}$ and $\tilde{M}$ are the average number of nodes and edges in each graph, respectively.

Datasets	#G	$\tilde{N}$	$\tilde{M}$	F	#classes
DD	1178	284.32	715.66	89	2
ENZYMES	600	32.63	62.14	21	6
PROTEINS	1113	39.06	72.82	4	2
NCI1	4110	29.87	32.3	1	2
IMDB-B	1000	19.77	96.53	5	2
IMDB-M	1500	13	65.94	5	3
REDDIT	2000	429.63	497.75	5	2
COLLAB	5000	74.49	2457.78	5	3

3.3 Node Classification on Homogeneous Graphs

Following the recommended evaluation protocol for node classification [31], we create ten splits for each dataset where there is no public split. We use the linear evaluation protocol to quantify the quality of the representations obtained from the SSGRL methods, where we first train each SSGRL method on each split with no labels. Then, for each split, we set 5% and 15% as training and validation splits used for model selection and 80% for testing using a Linear (Logistic) classifier. Model selection is carried out using only 1 of the ten splits, and for a fair comparison, it is done for all the baselines. We tune all the hyper-parameters using Bayesian optimization [2]. In addition, for the baselines, the size of the representation dimension, D, is 128; for our model, it is 64, since we concatenate $\mathbf{Z}_1$ and $\mathbf{Z}_2$. The reported results for the small datasets are the accuracy on the test set averaged over the ten splits. For two large datasets, Yelp and Reddit, we only report the ROC-AUC and accuracy using the publicly available single set of train, validation, and test sets. The configuration of the hyper-parameters of DSGRL and additional details for this experiment are presented in the appendix.

The results are reported in Table 3. DSGRL based on feature and topology augmentations is better than the baselines in almost all datasets. In addition, it is also comparable with the semi-supervised methods. To highlight the scalability of DSGRL, we evaluate it on large-scale datasets and report the results in Table 4. The self-supervised baselines throw an out-of-GPU memory error for the large datasets. So we include semi-supervised methods for reference, and not comparison.

3.4 Node Classification on Heterogeneous Graphs

Although DSGRL is primarily optimized for homogeneous graphs, it can easily be applied to heterogeneous graphs.

Recent studies [16, 22, 25, 27, 41, 44, 51] have generalized graph contrastive learning (GCL) to heterogeneous graphs. In these approaches, composite sequences of edge types (i.e., meta-paths) are hand-crafted for an underlying graph to express different possible semantics. For example, a meta-path "author-paper-author" refers to collaboration in a citation graph. Next, meta-path-based graph augmentations are designed for GCL. In doing so, the qualities of the learned representations and augmentation rely heavily on the chosen meta-paths that are typically domain-specific. DSGRL differs from this line of research, as no pre-defined data augmentation or domain knowledge are assumed.

As a demonstration, we choose a popular dataset commonly used to benchmark methods for heterogeneous GRL. This dataset, IMDB [10], has three node types, which are `movie`, `director`, and `actor`, and there are two undirected edge types, which are `movie-to-director` and `movie-to-actor`. There are 11,616 (4,278–`movie`, 2081–`director`, 5,257–`actor`) nodes and 17,106 (4, 278–`movie-to-director` and 12,828–`movie-to-actor`) edges. The task is to classify the movie nodes as one of the three classes (*Action*, *Comedy*, and *Drama*).

The only modification we need is, instead of a single parameter Θ, we use $\Theta = \{\theta_R\}$, where θ_R denote the model parameter specific to an edge type R. We use the same experimental setting and splits provided in [10]. That is, the movie nodes are split into training (400–9.35%), validation(400–9.35%), and testing (3,478–81.30%) nodes. For the linear evaluation, we only use the test set just as in [10] with different training rates, which are 20%, 40%, 60%, and 80%. For example, when using 20% for training, we will use 20% of the test set for training the linear classifier and the remainder (80%) for evaluating and reporting the performance of the learned representations. In Table 5, we report the F1-Score of our model and previous methods. We take the figures for the baselines from [10], and we see that DSGRL achieves better performance than unsupervised methods and is sometimes comparable to the semi-supervised ones. Although the paper's primary focus is not on heterogeneous graphs, this experiment highlights the potential of successfully applying a similar approach to this kind of graph. In future work, we shall address this with more experiments, including more baselines and datasets, and introduce a self-supervised learning technique that generalizes not only to homogeneous but also heterogeneous graphs, including knowledge graphs.

3.5 Graph Property Prediction

In this experiment, we closely follow the experimental protocol suggested for a fair comparison of GNNs in GPP [9]. Since they provide public splits [3], we use their split in our experiment. For each dataset, they provide ten splits, and each split contains a model selection and test splits. The model selection has training and validation splits. Similar to the NC experiment, we use the linear evaluation protocol and a similar model selection procedure. As the number of features for the datasets in this experiment is usually tiny, we also tune D for the baselines between 32 and 128 and for our method between 32 and 64. Since the social dataset does not have features, we use the degree profile as features. The configuration of the hyper-parameters of DSGRL and additional details for this experiment are also in the appendix.

The results are reported in Table 6, and we use the published results for semi-supervised methods. As shown in the table, DSGRL with feature augmentation is better than the baselines in almost all cases. Although the topology augmentation is comparable with the feature one for the social

[2] Bayesian optimization using OPTUNA: https://optuna.org/

[3] https://github.com/diningphil/gnn-comparison/tree/master/data_splits

Table 3. Results of the NC experiment for the small datasets. OOR corresponds to out-of-resource (GPU Memory) Random-F and Random-T are untrained variants of DSGRL based on feature and toplogy augmentions, respectively.

Methods		Datasets					
		MAG-CS	AmazonPhoto	PubMed	GitHub	WikiCS	Deezer
Ours	Random-F	80.2±0.1	82.33±0.1	75.5±0.1	82.5±0.1	63.4±0.3	55.4±0.2
	Random-T	69.8±0.1	80.6±0.3	71.4±0.1	78.8±0.1	63.6±0.5	56.0±0.2
	Feature	91.4±0.1	**90.6±0.1**	82.4±0.2	**84.9±0.5**	**74.2±0.1**	58.2±0.1
	Topology	**92.9±0.1**	89.7±0.1	**83.8±0.1**	83.0±0.3	73.6±0.1	**59.3±0.1**
Self-Supervised (Baselines)	DGI	91.1±0.2	89.0±0.6	78.6±0.5	79.1±0.7	73.6±0.4	55.2±0.6
	MVGRL	88.2±0.1	87.2±0.1	77.0±0.3	79.8±0.1	61.7±0.1	OOR
	GCA	91.0±0.4	86.0±1.1	**83.8±0.2**	OOR	72.9±0.6	OOR
	BGRL	90.7±0.3	90.3±0.5	82.4±0.4	81.3±0.4	73.8±0.7	58.2±0.7
Semi-Supervised (References)	GCN	91.7±0.3	92.0±0.4	85.4±0.4	84.1±0.3	76.7±0.6	59.7±0.6
	GAT	91.3±0.1	92.3±0.5	84.7±0.1	85.5±0.3	77.3±0.5	59.5±0.6
	GRAPHSAGE	91.6±0.3	92.4±0.4	84.5±0.4	84.6±0.4	77.4±0.6	61.9±0.6

Table 4. Results of the NC experiment for two of the large-scale datasets. We only include semi-supervised and scalable GNN architectures for this experiment as the full-batch ones do not fit in GPU memory. In addition, all the SSGRL baselines throw an out-of-memory error.

Methods	Datasets	
	Yelp (ROC-AUC)	Reddit (Accuracy)
CLUSTERGCN (semi)	78.2	95.3
GRAPHSAINT (semi)	75.6	95.7
PPRGO (semi)	77.7	91.8
DSGRL (Random-F)	72.6	82.3
DSGRL (Feature)	75.2	89.3

Table 5. Results of the NC experiment for heterogeneous graph.

Metrics	Train %	Unsupervised/Self-supervised (Baselines)				Ours		Semi-Supervised (References)		
		NODE2VEC [12]	ESIM [30]	METAPATH2VEC [8]	HEREC [32]	Random-F	DSGRL	GAT	HAN [42]	MAGNN [10]
Macro-F1	20	49.00	48.37	46.05	45.61	41.58	**53.14**	53.64	56.19	59.35
	40	50.63	50.09	47.57	46.80	44.84	**54.90**	55.50	56.15	60.27
	60	51.65	51.45	48.17	46.84	44.12	**56.25**	56.46	57.29	60.66
	80	51.49	51.37	49.99	47.73	45.20	**60.28**	57.43	58.51	61.44
Micro-F1	20	49.94	49.32	47.22	46.23	42.44	**53.35**	53.64	56.32	59.60
	40	51.77	51.21	48.17	47.89	45.57	**54.89**	55.56	57.32	60.50
	60	52.79	52.53	49.17	48.19	44.61	**56.32**	56.47	58.42	60.88
	80	52.72	52.54	50.50	49.11	45.55	**60.05**	57.40	59.24	61.53

datasets, it could perform better for chemical datasets. We have similar observations for the baselines, which also alter the topology. Note that even the untrained variants of DSGRL are strong competitors for these datasets. Corroborating the observation in [39], that is, what is lost in data augmentation is compensated by the strong inductive bias of GNNs. We believe that corrupting the topology of such datasets requires careful consideration.

4 Related Work

In general, there have been many frameworks for contrastive learning. Mostly, they differ in their data augmentation techniques and the architectures they choose to prevent collapse.

Data Augmentation Although there are well-established data augmentation techniques in the computer vision domain, this is not the case for the graph domain [14, 47]. Different heuristics, based on high-order networks, perturbation of topology and attributes have been proposed [3, 14, 18, 40, 48]. It is unclear what the relative benefit of these augmentation strategies is, and little is known regarding the relevance of each strategy concerning different downstream tasks. Recently, studies [35, 39, 47] have proposed learnable and contextual augmentation techniques [35, 39, 53]. However, these methods are restrictive because they either specify a set of graph data augmentation techniques so that the learning is choosing the correct technique, or they only learn to dropout edges through adversarial training. A more relevant study, i.e., SimGRACE for graph property classification, [45], has perturbed the model weights using Gaussian noise rather than perturbing the node features or the topology. However, a study [22] has shown that data augmentation in the graph space is more complicated than Gaussian distribution can capture. Therefore SimGRACE learns sub-optimal representations compared to the previously-mentioned data augmentation counterparts.

On the other hand, recent studies [16, 22, 25, 27, 41, 44, 51] have extended graph contrastive learning to heterogeneous networks. However, all these approaches have employed meta-paths to design graph augmentations. Therefore, their performance and the augmentation's quality itself are heavily conditioned on the quality of the manually-chosen meta-paths.

Our study differs from these lines of research, as no predefined data augmentations exist. Secondly, we propose a flexible framework to jointly learn topological or feature augmentation suitable for a given graph.

Table 6. Results for graph property prediction experiment. We report the classification accuracy of three groups of methods: Semi-Supervised (References), Self-Supervised (Baselines), and the variants of our method. Bold indicates the best-performing method.

Methods		Datasets							
		Chemical				Social			
		DD	NCI1	PROTEINS	ENZYMES	IMDB-B	IMDB-M	REDDIT	COLLAB
Ours	Random-F	77.0±2.9	67.2±1.9	74.4±3.7	36.0±5.7	70.7±4.0	46.5±3.6	76.7±2.1	66.3±2.7
	Ramdom-T	75.6±2.3	71.0±1.9	73.0±1.8	36.0±5.6	67.0±4.7	43.3±3.9	69.4±1.1	66.5±1.5
	Feature	**78.0±2.9**	75.0±2.2	**75.6±3.4**	**54.0±3.7**	**71.9±3.4**	**50.3±3.4**	**78.3±2.1**	69.6±1.4
	Topology	75.8±3.0	72.8±1.9	74.4±4.5	37.8±5.7	71.7±3.6	**50.3±3.3**	**78.3±2.3**	**70.0±2.1**
Self-Supervised (Baselines)	ADGCL	69.9±3.8	67.7±1.2	71.2±2.28	20.5±3.6	69.1±3.4	41.7±2.2	70.8±2.6	67.7±2.5
	GRAPHCL	76.4±2.7	**75.2±1.3**	73.7±5.0	25.3±6.0	**71.9±4.8**	47.2±4.1	78.1±1.9	69.24±1.1
	INFOGRAPH	74.8±4.0	73.4±2.1	73.8±3.6	30.1±5.1	71.5±2.5	47.8±4.0	73.5±2.9	64.1±1.2
Semi-Supervised (References)	DGCNN	76.6 -/+ 4.3	76.4±1.7	72.9±3.5	38.9±5.7	69.2±3.0	45.6±3.4	87.8±2.5	71.2±1.9
	DIFFPOOL	75.0±3.5	76.9±1.9	73.7±3.5	59.5±5.6	68.4±3.3	45.6±3.4	89.1±1.6	68.9±2.0
	ECC	72.6±4.1	76.2±1.4	72.3±3.4	29.5±8.2	67.7±2.8	43.5±3.1	OOR	OOR
	GIN	75.3±2.9	80.0±1.4	73.3±4.0	59.6±4.5	71.2±3.9	48.5±3.3	89.9±1.9	75.6±2.3
	GRAPHSAGE	72.9±2.0	76.0±1.8	73.0±4.5	58.2±6.0	68.8±4.5	47.6±3.5	84.3±1.9	73.9±1.7

Architectures The key difference between existing contrastive architectures arises from the need to prevent trivial solutions. To this end, existing studies often rely on negative sampling or contrastive terms [14,34,40,43,47,48,52,53]. However, as sampling truly contrastive terms are difficult, other studies have used asymmetric architectures to prevent trivial solutions. Initially proposed for CV [6,11], such methods [5,18,38] have empirically shown that asymmetric networks and a stop gradient operation are sufficient to prevent collapse. Although the asymmetric methods avoid explicit negative sampling, they are mainly engineering tricks.

Recent studies [1,3,49] have introduced principled approaches based on regularization. Compared to contrastive architectures with negative sampling, these methods used a principled approach to prevent collapse. However, in contrast, they do not require explicit negative sampling.

5 Conclusion and Discussion

This paper presents a novel data-driven self-supervised graph representation learning method called DSGRL. Unlike existing methods, DSGRL learns augmentation governed by the graph's inherent signal. We propose two complementary approaches, one based on learning high-order topology and another on learning feature augmentations. In both cases, augmentation is jointly learned with the graph representation.

We perform an extensive empirical evaluation using eight graph property predictions and another nine node classification datasets, including heterogeneous and homogeneous graphs, which are publicly available. We compare DSGRL against seven popular and SOTA baselines, three for graph property prediction and four for node classification experiments. Furthermore, in both experiments, we closely follow recommended protocols for a fair comparison and tuned the hyper-parameters of all the baselines. The overall results confirm that DSGRL surpasses the baseline SOTA approaches.

Among the graph property prediction datasets, 4 of them are chemical, and 4 are social datasets. For the social graphs, the empirical results show that both augmentation techniques produce comparable results. Whereas for the chemical graphs, the topological augmentation does not perform well. The latter is also the case for the baselines, which rely on perturbing the topology. This aligns with existing studies that argue against topological perturbation for such datasets [39]. We believe topological augmentations for chemical datasets require further careful investigation.

Last, we report on the untrained variants of the DSGRL. Our results show that even the untrained model is significantly better for the chemical datasets than some of the baselines. The latter is consistent with recent findings [39], which show that the strong inductive bias of GNNs tends to compensate for what is lost in the augmentation.

Acknowledgements

This project has received funding from the EU-H2020European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No [http://rais-itn.eu]MarieCurie813162813162: RAIS – Real-time Analytics for the Internet of Sports.

References

[1] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning, 2021.

[2] Mikhail Belkin and Partha Niyogi, 'Laplacian eigenmaps for dimensionality reduction and data representation', *Neural Computation*, **15**(6), 1373–1396, (2003).

[3] Piotr Bielak, Tomasz Kajdanowicz, and Nitesh V. Chawla. Graph barlow twins: A self-supervised representation learning framework for graphs, 2021.

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin, 'Unsupervised learning of visual features by contrasting cluster assignments', *CoRR*, **abs/2006.09882**, (2020).

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin, 'Emerging properties in self-supervised vision transformers', *CoRR*, **abs/2104.14294**, (2021).

[6] Xinlei Chen and Kaiming He, 'Exploring simple siamese representation learning', *CoRR*, **abs/2011.10566**, (2020).

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[8] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami, 'Metapath2vec: Scalable representation learning for heterogeneous networks', in *Proceedings of the 23rd ACM SIGKDD*, KDD '17, p. 135–144, New York, NY, USA, (2017). Association for Computing Machinery.

[9] Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. A fair comparison of graph neural networks for graph classification, 2020.

[10] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King, 'MAGNN: Metapath aggregated graph neural network for heterogeneous graph embedding', in *Proceedings of The Web Conference 2020*. ACM, (apr 2020).

[11] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning, 2020.

[12] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks, 2016.

[13] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2018.

[14] Kaveh Hassani and Amir Hosein Khas Ahmadi, 'Contrastive multi-view representation learning on graphs', *CoRR*, **abs/2006.05582**, (2020).

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick, 'Momentum contrast for unsupervised visual representation learning', (2019).

[16] Baoyu Jing, Chanyoung Park, and Hanghang Tong, 'Hdmi: High-order deep multiplex infomax', in *Proceedings of the Web Conference 2021*, pp. 2414–2424, (2021).

[17] Zekarias T. Kefato and Sarunas Girdzijauskas, 'Gossip and attend: Context-sensitive graph representation learning', in *14-th International Conference on Web and Social Media*, ICWSM'20, (2020).

[18] Zekarias T. Kefato and Sarunas Girdzijauskas, 'Self-supervised graph neural networks without explicit negative sampling', *CoRR*, (2021).

[19] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[20] Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning, 2019.

[21] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations, 2020.

[22] Qi Li, Wenping Chen, Zhaoxi Fang, Changtian Ying, and Chen Wang, 'A multi-view contrastive learning for heterogeneous network embedding', *Scientific Reports*, **13**(1), 6732, (2023).

[23] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[24] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann, 'Tudataset: A collection of benchmark datasets for learning with graphs', in *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, (2020).

[25] Chanyoung Park, Donghyun Kim, Jiawei Han, and Hwanjo Yu, 'Unsupervised attributed multiplex network embedding', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5371–5378, (2020).

[26] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena, 'Deepwalk: Online learning of social representations', in *Proceedings of the 20th ACM SIGKDD*, KDD'14, p. 701–710, New York, NY, USA, (2014). ACM.

[27] Yuxiang Ren, Bo Liu, Chao Huang, Peng Dai, Liefeng Bo, and Jiawei Zhang, 'Heterogeneous deep graph infomax', *arXiv preprint arXiv:1911.08538*, (2019).

[28] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding, 2021.

[29] Ahmed E Samy, Lodovico Giaretta, Zekarias T Kefato, and Šarūnas Girdzijauskas, 'Schemawalk: Schema aware random walks for heterogeneous graph embedding', in *Companion Proceedings of the Web Conference 2022*, pp. 1157–1166, (2022).

[30] Jingbo Shang, Meng Qu, Jialu Liu, Lance M. Kaplan, Jiawei Han, and Jian Peng. Meta-path guided embedding for similarity search in large-scale heterogeneous information networks, 2016.

[31] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation, 2019.

[32] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and Philip S. Yu. Heterogeneous information network embedding for recommendation, 2017.

[33] Fan-Yun Sun, Jordan Hoffman, Vikas Verma, and Jian Tang, 'Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization', in *International Conference on Learning Representations*, (2020).

[34] Mengying Sun, Jing Xing, Huijun Wang, Bin Chen, and Jiayu Zhou. Mocl: Contrastive learning on molecular graphs with multi-level domain knowledge, 2021.

[35] Susheel Suresh, Pan Li, Cong Hao, and Jennifer Neville. Adversarial graph augmentation to improve graph contrastive learning, 2021.

[36] Qiaoyu Tan, Sirui Ding, Ninghao Liu, Soo-Hyun Choi, Li Li, Rui Chen, and Xia Hu, 'Graph contrastive learning with model perturbation', (2023).

[37] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei, 'Line: Large-scale information network embedding', WWW'15, p. 1067–1077, Republic and Canton of Geneva, CHE, (2015). ACM.

[38] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Rémi Munos, Petar Veličković, and Michal Valko. Bootstrapped representation learning on graphs, 2021.

[39] Puja Trivedi, Ekdeep Singh Lubana, Yujun Yan, Yaoqing Yang, and Danai Koutra. Augmentations in graph contrastive learning: Current methodological flaws; towards better practices, 2021.

[40] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax, 2018.

[41] Can Wang, Sheng Zhou, Kang Yu, Defang Chen, Bolang Li, Yan Feng, and Chun Chen, 'Collaborative knowledge distillation for heterogeneous information network embedding', in *Proceedings of the ACM Web Conference 2022*, pp. 1631–1639, (2022).

[42] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu, 'Heterogeneous graph attention network', in *The World Wide Web Conference*, WWW '19, p. 2022–2032, New York, NY, USA, (2019). Association for Computing Machinery.

[43] Xiao Wang, Nian Liu, Hui Han, and Chuan Shi. Self-supervised heterogeneous graph neural network with co-contrastive learning, 2021.

[44] Xiao Wang, Nian Liu, Hui Han, and Chuan Shi, 'Self-supervised heterogeneous graph neural network with co-contrastive learning', in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 1726–1736, (2021).

[45] Jun Xia, Lirong Wu, Jintao Chen, Bozhen Hu, and Stan Z Li, 'Simgrace: A simple framework for graph contrastive learning without data augmentation', in *Proceedings of the ACM Web Conference 2022*, pp. 1070–1079, (2022).

[46] Yaochen Xie, Zhao Xu, Jingtun Zhang, Zhengyang Wang, and Shuiwang Ji. Self-supervised learning of graph neural networks: A unified review, 2021.

[47] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang, 'Graph contrastive learning automated', *CoRR*, **abs/2106.07594**, (2021).

[48] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen, 'Graph contrastive learning with augmentations', *CoRR*, (2020).

[49] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny, 'Barlow twins: Self-supervised learning via redundancy reduction', (2021).

[50] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method, 2020.

[51] Yanqiao Zhu, Yichen Xu, Hejie Cui, Carl Yang, Qiang Liu, and Shu Wu, 'Structure-enhanced heterogeneous graph contrastive learning', in *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pp. 82–90. SIAM, (2022).

[52] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning, 2020.

[53] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang, 'Graph contrastive learning with adaptive augmentation', in *Proceedings of WWW'21*, WWW '21, p. 2069–2080, New York, NY, USA, (2021). ACM.