Augmentations in Graph Contrastive Learning: Current Methodological Flaws & Towards Better Practices

Puja Trivedi
University of Michigan
pujat@umich.edu

Ekdeep Singh Lubana
University of Michigan
eslubana@umich.edu

Yujun Yan
University of Michigan
yujunyan@umich.edu

Yaoqing Yang
University of California, Berkeley
yqyang@berkeley.edu

Danai Koutra
University of Michigan
dkoutra@umich.edu

ABSTRACT

Unsupervised graph representation learning is critical to a wide range of applications where labels may be scarce or expensive to procure. Contrastive learning (CL) is an increasingly popular paradigm for such settings and the state-of-the-art in unsupervised *visual* representation learning. Recent work attributes the success of visual CL to use of task-relevant augmentations and large, diverse datasets. Interestingly, *graph* CL frameworks report strong performance despite using orders of magnitude smaller datasets and employing domain-agnostic graph augmentations (DAGAs). Motivated by this discrepancy, we probe the quality of representations learnt by popular graph CL frameworks using DAGAs. We find that DAGAs can destroy task-relevant information and harm the model's ability to learn discriminative representations. On small benchmark datasets, we show the inductive bias of graph neural networks can significantly compensate for this weak discriminability. Based on our findings, we propose several sanity checks that enable practitioners to quickly assess the quality of their model's learned representations. We further propose a broad strategy for designing task-aware augmentations that are amenable to graph CL and demonstrate its efficacy on two large-scale, complex graph applications. For example, in graph-based document classification, we show task-relevant augmentations improve accuracy up to 20%.

CCS CONCEPTS

• **Computing methodologies** → **Instance-based learning**.

KEYWORDS

Graph Neural Networks, Contrastive Learning, Data Augmentation

ACM Reference Format:
Puja Trivedi, Ekdeep Singh Lubana, Yujun Yan, Yaoqing Yang, and Danai Koutra. 2022. Augmentations in Graph Contrastive Learning: Current Methodological Flaws & Towards Better Practices. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3485447.3512200

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3512200

1 INTRODUCTION

Graph neural networks (GNNs) have been successfully used to learn representations for various supervised or semi-supervised graph-based tasks, including graph-based similarity search for web documents [18], fake news detection through propagation pattern classification [4, 12, 20, 44], activity analysis in web and social networks (e.g., discussion threads on Reddit, code repository networks on Github) [55], and scientific graph classification [25, 77, 84]. However, in many practical scenarios, labels are scarce or difficult to obtain. For example, web pages are seldom assigned with labels which summarize their contents, labeling fake news can be time-consuming, and labeling drugs according to their toxicity requires expensive wet lab experiments or analysis [13, 28, 29, 96]. *Contrastive learning* (CL) is an increasingly popular unsupervised graph representation learning paradigm for such label scarce settings [15, 22, 62, 86, 86, 87] and is currently the state-of-the-art in unsupervised visual representation learning [6–8, 24].

Broadly, CL frameworks learn representations by maximizing similarity between augmentations of a sample (positive views) while simultaneously minimizing similarity to other samples in the batch (negative views). Recent theoretical and empirical works attribute the impressive success of visual CL (VCL) to two key principles: (i) leveraging *strong, task-relevant data augmentation* [52, 68, 76, 79, 95] and (ii) training on *large, diverse datasets* [3, 8, 24, 47, 51]. By using appropriate data augmentations, VCL frameworks learn high quality representations that are *invariant* to properties *irrelevant* to downstream task performance; thereby preserving task-relevant properties and preventing the model from learning brittle shortcuts [8, 52, 54, 68]. Large, diverse datasets are necessary as VCL frameworks routinely use 1K–8K samples in a batch to ensure that enough negative views are available to train stably [6, 8, 10, 24]. Representations learnt using VCL and self-supervised learning in general have been found to be more robust [26], transferable [30] and semantically aligned [57] than their supervised counterparts.

Interestingly, graph CL (GCL) frameworks often deviate from these key principles and yet report seemingly strong task performance. Small, binary graph classification datasets [45] are routinely used to benchmark GCL frameworks. Moreover, due to the non-euclidean, discrete nature of graphs, it can be difficult to design task-relevant graph data augmentations [38, 93] or know what invariances are useful for the downstream task. Therefore, frameworks often rely upon domain-agnostic graph augmentations (DAGAs) [87]. However, DAGAs can destroy task relevant information

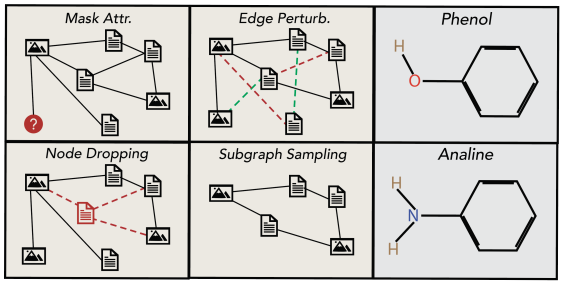


Figure 1: [Left] *Domain-Agnostic Graph Augmentations (DAGAs)* introduced in [87]. Deletion/addition in red/green. [Right] *False Positive Samples.* Acidic molecule Phenol and basic molecule Analine are structurally similar but have different properties. DAGAs can inadvertently generate this pair as a positive view, resulting in *similar* representations for semantically *dissimilar* entities.

and yield *invalid/false positive* samples (see Fig. 1). It is also unclear if DAGAs induce invariances that are useful or semantically meaningful with respect to the downstream task.

In this work, we investigate the implications of the aforementioned discrepancies by probing the quality of representations learnt by popular GCL frameworks using DAGAs. We show that DAGAs can destroy task-relevant information and lead to weakly discriminative representations. Moreover, on popular, small benchmark datasets, we find that flawed evaluation protocols and the strong inductive bias of GNNs mitigate limitations of DAGAs. Our analysis offers several actionable sanity checks and better practices for practitioners when evaluating GCL representation quality. Further, through two case studies on larger, more complex datasets, we demonstrate that task-aware augmentations (TAAs) are necessary for strong performance and discuss how to identify such augmentations amenable to GCL. Our main contributions are summarized as follows:

- **Analysis of limitations in domain-agnostic augmentations:** We demonstrate that commonly-used DAGAs lead models to learn weakly discriminative representations by inducing invariances to invalid views or false-positives. Across several architectures and datasets, we find these shortcomings are mitigated by the strong inductive bias of GNNs, which allow existing methods to achieve competitive results on benchmark datasets.
- **Identification of methodological flaws & better practices**: We contextualize recent theoretical work in visual self-supervised learning to identify problematic practices in GCL: (i) the use of small datasets and (ii) training with negative-sample frameworks on binary classification datasets. Furthermore, we provide carefully-designed sanity checks for practitioners to assess the benefits of proposed augmentations and frameworks.
- **Case studies with strong augmentations:** In two case studies on different data modalities, we demonstrate how to leverage simple domain knowledge to develop strong, task-aware graph augmentations. Our systematic process results in up to 20% accuracy improvements.

For reproducibility, our code and data are available at https://github.com/GemsLab/GCLAugmentations-BestPractices.

2 PRELIMINARIES & RELATED WORK

We begin by introducing CL. We then discuss how strong, task-relevant augmentations and large, diverse datasets underpin the success of VCL. Finally, GCL and graph data augmentation are discussed. Please see Appendix D for additional related work.

2.1 Contrastive Learning (CL)

Frameworks & Losses. Several CL frameworks [8, 21, 24] have been proposed to enforce similarity between positive samples and dissimilarity between negative samples, where positive samples are generated through data augmentation. Normalized temperature-scaled cross entropy (NT-XENT) is a popular objective used by several state-of-the-art CL frameworks [8, 61, 63, 71, 80, 86, 87] and is defined as follows. Let $\mathcal{X}$ be a data domain, $\mathcal{D} = \{x_{[1...n]} | x_i \in \mathcal{X}\}$ be a dataset, $\mathcal{T}: \mathcal{X} \rightarrow \tilde{\mathcal{X}}$ be a stochastic data transformation that returns a positive view, and $f: \{\mathcal{X}, \tilde{\mathcal{X}}\} \rightarrow \mathbb{R}^d$ be an encoder. Further, assume we are given a batch of size N, similarity function $\text{sim}: (\mathbb{R}^d, \mathbb{R}^d) \rightarrow [0, 1]$, temperature parameter τ, and encoded positive pair $\{z_i, z_j\}$. Then, NT-XENT can be defined as:

$$\ell_{i,j} = -\log \frac{\exp\left(\text{sim}\left(z_i, z_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp\left(\text{sim}\left(z_i, z_k\right)/\tau\right)}. \quad (1)$$

Here, the numerator encourages the positive pair to be similar, while the denominator encourages negative pairs ($k \neq i$) to be dissimilar. Alternative CL objectives may enforce such (dis)similarity differently (e.g., through margin maximization [70] or cosine similarity [21]), but the principles discussed below uniformly explain the success of contrastive learning frameworks [2].

The role of augmentations. Recent work [52, 68, 69] has demonstrated that data augmentation is critical for training CL frameworks. Theoretically, Tian et. al [68] show that positive views should preserve task-relevant information, while simultaneously minimizing task-irrelevant information [69]. Training on such views introduces invariances to irrelevant information, leading to more generalizable representations. Indeed, state-of-the-art VCL frameworks [6, 8, 9, 17, 24] rely upon strong, task relevant data augmentation to generate such views. For example, Purushwalkam et al. [52] show that augmentations used by SimCLR introduce "occlusion invariance", which is useful in classification tasks where objects may be occluded. Overall, we highlight that augmentation strategies are not universal [86, 87] and must align with the task; e.g., semantic segmentation tasks would benefit more from augmentations that induce view-point invariances [52].

The role of large, high-quality datasets. Empirically, CL frameworks [8, 10, 24] often require many negative samples in each batch to avoid class collisions (i.e., false positives) [2]. Further, recent theoretical work has shown that optimizing Eq. (1) is equivalent to learning an estimator for the mutual information shared between positive views, where the quality of this estimate is upper-bounded by batch-size [51, 71]. These properties combine to necessitate the use of large, diverse datasets in contrastive learning.

2.2 Graph Contrastive Learning (GCL)

Frameworks. In this paper, we focus on three state-of-the-art unsupervised representation learning frameworks for graph classification that represent different methodological perspectives: GraphCL [87], InfoGraph [62] and MVGRL [22]. Similar to SimCLR, GraphCL uses NT-XENT to contrast representations of augmented samples using a shared encoder. Much like DeepInfoMax [27], InfoGraph maximizes the mutual information between local and global views, where corresponding views are obtained through subgraph sampling and graph-pooling. Meanwhile, MVGRL mirrors CMC [67] and uses dual encoders to contrast multiple views of a graph, where views are generated by first running a diffusion process (e.g. Personalized Page Rank [48], Heat Kernel [37]) over the graph and then sampling subgraphs.

Graph data augmentation. Existing GCL frameworks leverage three main strategies to generate views: feature or topological perturbation (GraphCL), sampling (InfoGraph), and/or diffusion processes (MVGRL). We focus on the domain-agnostic graph augmentations (DAGAs) introduced by GraphCL, shown in Fig. 1, as these are more popular in recent frameworks [65, 86, 87], composable [8, 17], fast, and do not require dual view encoders. An empirical study on the benefits of DAGAs in GCL [87] demonstrates that (i) composing augmentations and adjusting augmentation strength to create a more difficult instance discrimination task improves downstream performance and (ii) augmentation utility is dataset dependent. However, a critical assumption underlying DAGA is that by limiting augmentation strength such that only a fraction of the original graph is modified, task-relevant information is not significantly altered. In Sec. 3, we revisit this assumption to show that it does not hold for many datasets and discuss the implications of training with poorly augmented graphs. Clearly, it is expected that models trained with task-aware augmentations (TAAs) that induce useful invariances will learn better features than those trained with DAGAs. However, graphs are often used as abstracted representations of structured data, such as molecules [96] or point clouds [59], and it is often unclear how to represent task-relevant invariances after abstracting to the graph space. In Sec. 4, we discuss a broad strategy for identifying augmentations that induce task-relevant invariances in the abstracted, graph space and demonstrate the significant performance boosts achieved by using such augmentations.

Automated Graph Data Augmentation. Concurrent works [23, 32, 33, 40, 50, 63, 86, 89] have begun investigating *automated* graph data augmentation as a means of both avoiding costly trial and error when selecting augmentations and generating more informative, task relevant views. These methods often use bi-level optimization objectives and/or viewmakers [64] to jointly learn representations and augmentations (cf. Appendix D for more details). Our analysis (Sec. 3) remains pertinent for GCL with automated augmentations. Namely, the proposed sanity checks are not augmentation specific, the identified evaluation flaws must still be considered, and untrained models should still be included as baselines. Also, our discussion on the benefits and properties of TAAs (Sec. 4) remains relevant as it is difficult to identify post-hoc if an automated augmentation strategy is inducing semantically meaningful invariances or exploiting shortcuts.

3 REVISITING AUGMENTATIONS & EVALUATION IN GCL

In this section, we investigate how existing GCL frameworks deviate from the principles underlying the success of VCL methods and the effects of such deviations. We discuss and establish three key observations:

- **(O1)** Standard graph data augmentation is susceptible to altering graphs semantics and task-relevant information.
- **(O2)** Training on such augmentations can lead to weakly discriminative representations.
- **(O3)** The strong inductive bias of randomly-initialized GNNs obfuscates the performance of weak representations and misaligned evaluation practices.

Empirical Setup. In our analysis, we focus on commonly used graph classification datasets (Table 1) [45]. Official implementations for GraphCL[1], InfoGraph[2], and MVGRL[3] are used. We consider the encoder architecture used by [87] and report results with graph convolutional layers from GIN [82] (original implementation), PNA [11], SAGE [19], GAT [73], and GCN [35]. See Appendix A for details on the training setup.

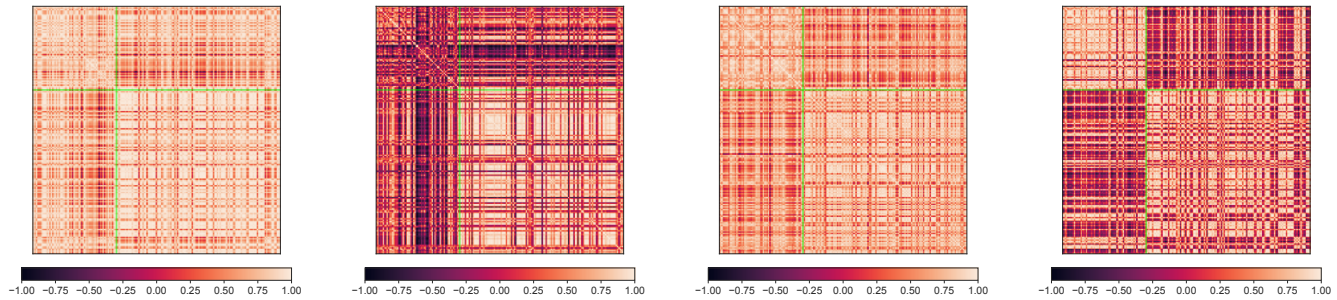
Table 1: *Dataset Description*

Name	Graphs	Classes	Avg. Nodes	Avg. Edges	Domain
IMDB-BINARY [85]	1000	2	19.77	96.53	Social
REDDIT-BINARY [85]	2000	2	429.63	497.75	Social
GOSSIPCOP [60]	5464	2	55.48	54.51	News
DEEZER [55]	9629	2	23.49	65.25	Social
GITHUB SGZR [55]	12725	2	113.79	234.64	Social
MUTAG [39]	188	2	17.93	19.79	Molecule
PROTEINS [5]	1113	2	39.06	72.82	Bioinf.
DD [58]	1178	2	284.32	715.66	Bioinf.
NCI1 [77]	4110	2	29.87	32.30	Molecule

3.1 (O1) Domain-agnostic graph augmentations alter task-relevant information

Given the importance of data augmentation in representation learning, several works [52, 68, 76, 81, 95] have investigated its properties. Recently, Gontijo-Lopes et al. [42] identified an empirical trade-off when selecting amongst augmentations to improve model generalization. Intuitively, augmentations should generate samples that are close enough to the original data to share task-relevant semantics and different enough to prevent trivially similar samples. This trade-off can be quantified through two metrics, *affinity* and *diversity*. Affinity measures the distribution shift between the augmented and original sample distributions. Diversity quantifies how difficult it is to learn from augmented samples instead of only training samples [42]. While augmentations that best improve generalization optimize for both metrics [42], it is not clear that DAGAs also optimize for both. For example, molecular graph classification tasks are commonly used to evaluate GCL frameworks. However, as noted in Fig. 1, limited perturbations are needed to invalidate a molecule or significantly alter its function. Here, augmented data is sufficiently diverse, but it is not clear if creating invalid molecule

[1] https://github.com/Shen-Lab/GraphCL
[2] https://github.com/fanyun-sun/InfoGraph
[3] https://github.com/kavehhassani/mvgrl



(a) Random Init. (85.76 ± 7.38) **(b) GraphCL** (86.80 ± 1.34) **(c) InfoGraph** (89.01 ± 1.13) **(d) MVGRL** (89.70 ± 1.1)

Figure 2: *Representational Similarity.* The normalized cosine similarity between all-pairs of representations is shown above for the MUTAG dataset. The on-diagonal blocks (indicated by green lines) show intra-class similarity, while off-diagonal blocks show inter-class similarity. MVGRL, which uses diffusion-based views, learns representations that have high *intra*-class similarity and low *inter*-class similarity, as desired. InfoGraph, which directly maximizes mutual information between local/global views, preserves high intra-class similarity, and has moderate inter-class similarity. GraphCL, which uses domain-agnostic graph augmentations, has low intra-class similarity in the upper left block. This indicates that training on false positive/invalid samples can negatively impact representational power.

Table 2: *Augmentation Affinity.* Affinity [42], measured by the difference between original and augmented accuracy of a supervised model, captures how much the data distribution has changed as a result of augmentation. We see that DAGAs lead to low affinity. This is expected for molecular datasets, where it is easy to create invalid molecules, and is also true for some social network datasets.

Dataset	Clean Train Acc.	Aug. Train Acc.
MUTAG	90.14 ± 1.36	37.67 ± 1.48
PROTEINS	70.70 ± 4.30	56.54 ± 8.11
NCI1	75.55 ± 4.60	60.15 ± 0.069
DD	84.06 ± 8.81	65.41 ± 14.87
REDDIT-BINARY	85.56 ± 3.21	50.56 ± 0.09
IMDB-BINARY	70.93 ± 0.046	50.11 ± 0.384
GOSSIPCOP	98.047 ± 0.37	96.03 ± 1.57

samples also leads to low affinity, indicating that task-relevant information have been destroyed. We conduct the following experiment to understand the affinity of DAGAs on benchmark datasets.

Experimental setup. We measure affinity as follows: (i) train a supervised PNA encoder on the original training data, (ii) generate an augmented dataset by using random node/subgraph dropping at 20% of the graph size, as suggested by [87] and (iii) evaluate on clean *and* augmented training data separately. The difference between clean and augmented accuracy quantifies the distribution shift induced by augmentations [42].

Hypothesis. We argue that while it is not expected that accuracy on augmented data will match that of clean data, augmented accuracy should be nontrivial if augmentations are indeed information-preserving [76, 79].

Results. In Table 2, we see a considerable difference between clean and augmented accuracy across datasets. This implies low affinity, i.e., a large shift between augmented and training distributions, and confirms that DAGAs can destroy task-relevant information. Consequently, training on such samples will harm downstream task performance, as shown by prior works on VCL [79] and elucidated below for GCL.

3.2 (O2) Domain-agnostic augmentations induce weak discriminability

Recall that contrastive losses maximize the similarity between representations of positive pairs while simultaneously minimizing the similarity amongst representations of negative samples. However, Obs. (**O1**) identifies that DAGAs have low affinity, which suggests that task-relevant information has been significantly altered. This implies that representation similarity will be maximized for samples that are *not* semantically similar, e.g., false positive samples. Consequently, the resulting representations may not be discriminative with respect to downstream classes—i.e., *intra*-class samples may have lower similarity than *inter*-class samples, counter to what is expected. This claim is investigated in the following experiment.

Experimental setup. We measure the discriminative power of representations learned using GCL as follows: given models trained using GraphCL, InfoGraph and MVGRL, we extract representations for the entire dataset. Then, we calculate cosine similarity between all representation pairs. Representational similarity from an untrained model is also included.

Hypothesis. If a model has learned discriminative representations, intra-class similarity should be high while inter-class similarity should be low.

Results. In Fig. 2, we plot the normalized cosine similarity between representations (sorted by class label), such that the upper left and lower right quadrants correspond to the similarity between same-class representations. Results on additional datasets can be found in Appendix A. We see that MVGRL (Fig. 2d) and InfoGraph (Fig. 2c) are less likely to encounter false positive pairs as they, respectively, use diffusion-based views and maximize mutual information over sampled subgraphs. GraphCL, which uses DAGAs, is more likely to encounter false positive samples that can harm discriminative power (Obs. (**O1**)). Correspondingly, MVGRL and InfoGraph both

Table 3: *Inductive Bias on Benchmark Datasets.* Following the same evaluation protocol as [62], we generate embeddings from an untrained N-Layer GIN encoder and perform classification using an SVM classifier. Results for GraphCL and InfoGraph are reported from [87]. Best accuracy is in bold; other models whose accuracy with standard deviation falls within the standard deviation of the best accuracy are underlined. We see across all datasets that untrained models have a strong inductive bias. On PROTEINS, DD, MUTAG DEEZER and GITHUB-SGZR, untrained models perform competitively against trained models.

Dataset (# Samples)	Random Init (3 layers)	Random Init (4 layers)	Random Init (5 layers)	GraphCL [87]	InfoGraph [62]
IMDB-BINARY (1000)	67.22 ± 7.77	61.26 ± 7.01	60.43 ± 5.92	71.14 ± 0.44	**73.03 ± 0.87**
REDDIT-BINARY (2000)	72.34 ± 6.64	64.57 ± 8.03	67.32 ± 7.41	**89.53 ± 0.84**	82.50 ± 1.42
DEEZER (9629)	**56.59 ± 0.01**	54.99 ± 1.74	54.87 ± 2.60	56.19 ± 0.015	55.89 ± 0.88
GITHUB SGZR (12725)	64.51 ± 0.05	64.93 ± 0.04	64.93 ± 0.89	**65.81 ± 0.413**	Out of Time
MUTAG (188)	85.76 ± 7.38	86.36 ± 6.51	86.73 ± 10.33	86.80 ± 1.34	**89.01 ± 1.13**
PROTEINS (1113)	73.64 ± 5.464	**74.46 ± 4.09**	74.22 ± 2.85	74.39 ± 0.45	74.44 ± 0.31
DD (1178)	73.23 ± 8.25	72.15 ± 7.25	77.08 ± 4.18	**78.62 ± 0.40**	72.85 ± 1.78
NCI1 (4110)	70.65 ± 1.99	70.36 ± 3.11	70.49 ± 2.42	**77.81 ± 0.41**	76.20 ± 1.06

learn representations with higher intra-class similarity than inter-class similarity. In contrast, GraphCL has low intra-class similarity as can be seen in the upper-left quadrant (Fig. 2b). This implies that the model has not learned features that capture the semantic similarity between the samples belonging to this class. However, we note that while MVGRL has learned discriminative representations, it requires dual encoders and it is unclear what invariances are learnt by training with diffusion-based views. Finally, we find that even though the randomly initialized, untrained model (Fig. 2a) has higher absolute values for average intra- and inter-class similarities than trained methods, it achieves inter-class similarity relatively lower than intra-class similarity, as required for discriminative applications. We further elaborate on this point in the next section.

Proposed evaluation practice. Given that CL frameworks directly optimize the similarity between representations, we argue that plotting representational similarity can serve as a simple sanity check for practitioners to assess the quality of their model's learned representations. Indeed, models are often only assessed through linear evaluation or task accuracy, which may hide differences in the discriminative power of representations. For example, as shown in Fig. 2, InfoGraph and MVGRL have similar task accuracy, but MVGRL has learnt more discriminative representations.

Having established that DAGAs can lead to invalid or false positive augmented samples and that training on such samples can lead to poorly-discriminative representations, we next investigate whether other factors are bolstering GCL performance. Specifically, we discuss the role of randomly initialized, untrained GNN inductive bias and identify flaws in current GCL evaluation practices.

3.3 (O3) Strong inductive bias of random models reduces GCL inefficiencies

As noted in Obs. (**O2**), randomly-initialized, untrained GNNs can produce representations that are already discriminative without any training (Fig. 2a). While the strength of inductive bias of GNNs in (semi-) supervised settings has been noted before [35, 56, 63, 91], we aim to better contextualize the performance of GCL frameworks by conducting a systematic analysis of the inductive bias of GNNs, using several datasets and architectures. Understanding the performance of untrained models helps contextualize the cost of training.

Empirical setup. For DEEZER and GITHUB-SGZR, a PNA encoder is used to stabilize training. All other datasets are trained with a GIN encoder. MVGRL ran out-of-memory so we did not include it in this evaluation. See Appendix A for more details.

Results. As shown in Table 3, randomly-initialized, untrained models perform competitively against trained models on several benchmark datasets. It is likely that some of the negative effects of training with DAGAs (Obs. (**O1**)–(**O2**)) were mitigated by this strong inductive bias. However, note that it becomes difficult to justify the additional cost of GCL on datasets where task performance and representation quality are not noticeably better than untrained models. Below, we discuss how to fairly evaluate GCL frameworks and how popular benchmark datasets are, in fact, inappropriate for GCL.

Proposed evaluation practices. Given that randomly-initialized, untrained models are a non-trivial baseline for GCL frameworks, we argue that they should be included when evaluating novel frameworks to contextualize the benefits of unsupervised training. While some recent works [63, 83] include untrained models in their evaluation, this practice remains far from standardized.

Furthermore, CL frameworks often define negative samples through the other samples in the batch. Given the limited size of popular benchmark datasets (Table 3), it can be difficult to ensure that each batch is large enough to train stably. Further, given that these benchmarks are often binary classification tasks, half the samples, in a balanced setting, are expected to share the positive pair's label but be treated as negative samples. This implies that representations learned with GCL may not be discriminative because models have minimized similarity for semantically related examples. We thus argue that evaluating *GCL* frameworks on these datasets is flawed and this practice should be discontinued.

We highlight that Dwivedi et al. [14] also find popular graph classification datasets are problematic in general, but for the specific case of GraphCL, this point is of some urgency as such small-scale datasets are part of standard GCL evaluation [63, 86]. However, we note that self-supervised frameworks that do not rely on negative samples, such as BYOL[17] and SimSiam[9], can be used as

an appropriate alternative for binary datasets. Such frameworks maximize similarity between sample augmentations and avoid degenerate solutions via stop-gradient operations and exponentially moving average target networks.

3.4 Summary of Proposed Evaluation Practices

We summarize the practices that we hope will be adopted in future graph CL research:

- Given that DAGA can destroy task-relevant information and harm the model's ability to learn discriminative representations, there is need for designing context-aware graph augmentations (Sec. 4).
- Randomly initialized, untrained GNNs have strong inductive bias and should be reported during evaluation.
- Small, binary graph datasets are inappropriate for evaluating GCL frameworks.
- GCL frameworks should be comprehensively evaluated using metrics beyond accuracy to assess representation quality.

4 BENEFITS & DESIGN OF TASK-AWARE AUGMENTATIONS

In this section, we exemplify the benefits of adhering to key VCL principles by defining a broad strategy for finding task-aware augmentations in scenarios where prior domain knowledge is available. We note the goal of this strategy is not to resolve problematic data augmentations in GCL. Instead, we use the proposed strategy to help elucidate the benefits of abiding by VCL principles in two careful case studies.

Augmentation strategy: For many graph-based representation learning tasks, structured data, such as documents [46], propagation patterns [44], molecules [28], maps [31], and point-clouds [59], are first abstracted as graphs via a deterministic process before task-specific learning can begin. In this practical setting, our idea is to leverage knowledge pertaining to the original, structured data to find augmentations that will, in the abstracted graph space, (i) preserve task-relevant information, (ii) break view symmetry, and (iii) introduce semantically meaningful invariance. In our first case study, which focuses on a graph-based document classification task, we achieve this goal by exploiting existing natural language augmentations [78] and directly perturbing the raw input before its graph is constructed. However, when given a sufficiently complex graph construction process, it can be unclear if augmentations in the original space will induce useful invariances or retain task-relevant information in the abstracted graph space. In our second case study, which focuses on image classification using super-pixel nearest-neighbor graphs, we encounter this setting and propose to avoid destruction of task-relevant information by deliberately introducing task-*irrelevant* information. We then use augmentations designed to induce invariance to such *irrelevant* information.

4.1 Case Study 1: Document Classification

We first focus on a binary graph-based document classification task. As shown by prior work [18], graph-based representations are effective at capturing not only the content but also the structure of a document, leading to improved classification performance in this setting. Here, our goal is to demonstrate adhering to VCL principles by using TAAs is needed to improve task performance.

Dataset & Task. The task is to classify movie reviews and plot summaries according to their subjectivity. Following [46], we convert the Subjectivity document dataset [49] (10k samples) into co-occurrence graphs, where nodes represent words, edges indicate that two words have co-occurred within the same window (e.g. window size 2 and 4), and node features are word2vec [43] embeddings. An example of this conversion is shown in Fig. 3a. Note that we only use positive-view-based self-supervised learning frameworks (e.g., SimSiam, BYOL) because this is a binary classification task (see Sec 3.3). Accuracy is computed using a kNN classifier.

Setup of GNN models. We use a Message Passing Attention Network [46] as the encoder, and a 2-layer MLP as the predictor. The representation dimension is 64, and models are trained using Adam [34] with LR=5e-4. Additional training details are given in Appendix B. We report results with the original GCN layer used by [46], as well as with GraphSAGE [19] and GIN [82] layers replacing it.

Domain-Agnostic Graph Augmentations. We conduct an informal grid search to select which DAGAs and augmentation strengths to use. Among node, edge, and subgraph dropping at {5%, 10%, 20%} of text length, we find generating both views using subgraph dropping (10%) performs the best. Generating one view with subgraph dropping (10%) and the other with node-dropping (10%) performs second best. We evaluate both strategies.

Task-Aware Augmentations. Recently, Wei et al. [78] proposed several intuitive augmentations for use in natural language processing, namely: synonym replacement, random word insertion, random word swapping and random word deletion, where the augmentation strength is determined by the sentence length. (See Fig. 3b for an example.) By design, these augmentations introduce invariances that are useful to downstream tasks (e.g., invariance to the occasional dropped word), preserve task-relevant information, and break view symmetry in the natural language modality. Due to a co-occurrence based construction process, changes in the underlying document will manifest in the corresponding graph, so it is likely that augmentations remain effective for the abstracted space.

Results. As shown in Table 4, task-relevant, natural language augmentations perform considerably better (up to +20%) than domain agnostic graph augmentations for both window sizes. Notably, TAAs are necessary to significantly improve performance over an untrained baseline, indicating that adhering to key principles of VCL is indeed beneficial.

Potential Graph Space Augmentations. While natural language augmentations modify samples prior to the graph construction process, it is easy to see that they can be converted into graph augmentations, effectively infusing DAGAs with domain knowledge on how to perturb co-occurence graphs. Specifically, synonym replacement is equivalent to replacing node features of the selected word (node) with the closest word2vec embedding. Random insertion can be approximated in the co-occurence graph by (i) creating a new node with a randomly selected word2vec embedding and (ii) duplicating the connections of an existing node. Random deletion can be represented by (i) randomly removing a node and (ii) rewiring the

(a) Original **(b) Natural Language Space** **(c) Graph Space**

Figure 3: *Augmentations for Document Classification:* Documents are represented as co-occurrence graphs [46, 92], where words are treated as nodes with word2vec embeddings, and edges indicate co-occurrence in a sliding windows [43]. As shown in (b), we perform synonym replacement (purple) and random word insertion (green) to augment sentences without losing task-relevant information [78]. In (c), we show random node (word) deletion (red). Our results show that natural language space augmentations improve classification accuracy substantially over baseline augmentations.

Table 4: *Document Classification.* We use domain-agnostic subgraph dropping (S) and node-dropping (N) at 10% and 5% of sentence length, respectively, for baseline augmentations. For task-aware augmentations, we stochastically apply synonym replacement (5%), random insertion (5%), random swapping (5 %) and random deletion (10 %). Random Accuracy with window-size = 2 is 58.46 ± 1.97**. Random Accuracy with window-size = 4 is** 63.93 ± 0.045**.**

	GCN		SAGE		GIN	
Augmentation	**SimSiam Acc.**	**BYOL Acc.**	**SimSiam Acc.**	**BYOL Acc.**	**SimSiam Acc.**	**BYOL Acc.**
S. vs. S (ws =2)	69.41 ± 7.28	62.98 ± 3.12	59.17 ± 8.36	67.17 ± 2.70	55.67 ± 4.61	65.02 ± 2.00
S vs. N (ws =2)	57.84 ± 4.31	65.78 ± 8.22	56.74 ± 1.70	63.77 ± 2.90	58.2 ± 8.24	74.26 ± 3.80
Context-Aware (ws = 2)	**83.65 ± 2.31**	**78.12 ± 2.73**	**81.28 ± 2.54**	**78.23 ± 4.53**	**80.37 ± 4.07**	**77.79 ± 0.09**
S vs. S (ws = 4)	61.76 ± 5.12	66.38 ± 2.29	54.68 ± 1.53	67.37 ± 1.11	54.71 ± 3.00	66.18 ± 2.34
S vs. N (ws = 4)	55.38 ± 1.99	68.311.88	59.23 ± 8.03	70.6 ± 4.85	53.31 ± 1.36	66.59 ± 1.57
Context-Aware (ws = 4)	**81.12 ± 3.97**	**74.05 ± 5.465**	**80.67 ± 10.36**	**75.65 ± 5.54**	**75.30 ± 15.61**	**76.55 ± 7.43**

modified graph to connect neighbors of the removed node. Random swap is equivalent to swapping the features of two nodes. We highlight that domain-agnostic subgraph and node dropping do *not* rewire the co-occurence graph. Thus, it is unclear what invariance to these augmentations represents in the original data modality. In Appendix B, we show that graph-space and document-space synonym replacement perform comparably, but leave the evaluation of other converted graph space augmentations to future work.

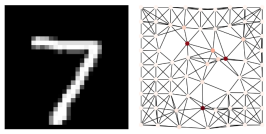
In this case study, we were able directly leverage augmentations in the original modality, which are known to preserve task-relevant information and induce useful invariances, to significantly outperform DAGAs. The next case study focuses on a more challenging setting where augmentations in the original modality are not immediately amenable to GCL due to a complex graph construction process and GNN architectural invariances.

4.2 Case Study 2: Super-pixel Classification

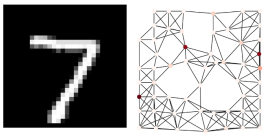
Our second case study is based on super-pixel MNIST classification, a standard benchmark for evaluating GNN performance [14, 36]. Here, we pursue an alternative strategy for task-aware augmentation where augmentations must induce invariance to deliberately irrelevant information (e.g., color for digit classification).

Dataset & Task. We follow the established protocols in [14, 36] to create super-pixel representations of MNIST, where each image is represented as a k-nearest neighbors graph between super-pixels (homogeneous patches of intensity). Nodes map to super-pixels, node features are super-pixel intensity and position, and edges are connections to k neighbors. An example is shown in Fig. 4.

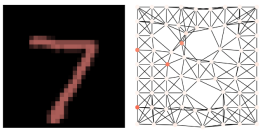
Setup of GNN models. The following architecture is used for experiments. The encoder is 5-layer GIN architecture similar to [87] and [14]. The projector is a 2-layer MLP and there is no predictor. Models are trained for 80 epochs, using Adam [34] with LR of 1e-3, and the representation dimension is set to 110. The models are trained using SimSiam [9], BYOL [17], and SimCLR [8]. We give more training details in Appendix C. While composing augmentations is known to improve performance on vision tasks, we avoid it here in order to fairly compare to graph baselines, which only consider a single augmentation.



(a) Original



(b) Node Dropping



(c) Colorizing

Figure 4: *Augmentations for Super-pixel Classification.* Node dropping alters graph topology and it is unclear if task-relevant information is preserved. Colorizing preserves task-relevant information by only perturbing node features.

Table 5: *Super-pixel Classification.* KNN Accuracy after unsupervised training with Node Dropping (ND) or context aware graph augmentations (Colorize) is reported. Context aware augmentations improve performance. Accuracy of randomly initialized model is 37.79 ± 0.03**.**

Aug.	SimSiam Acc.	SimCLR Acc.	BYOL Acc.
ND (20%)	66.30 ± 0.33	68.56 ± 0.16	**65.32 ± 0.95**
ND (30%)	61.30 ± 0.48	68.07 ± 0.37	61.87 ± 1.03
Colorize	**68.95 ± 1.20**	**73.67 ± 0.10**	64.42 ± 2.385

Domain-Agnostic Graph Augmentations. Following [87], we apply random node dropping at 20% of the graph size to obtain both samples in the positive pair.

Task-Aware Augmentations. While geometric image augmentations [8], such as horizontal flipping and rotating, generally preserve task-relevant information and introduce semantically meaningful invariance, they cannot break view symmetry in GCL frameworks as GNNs are permutation invariant. Therefore, the representations of a pair of flipped images will be similar as their corresponding super-pixel graph representations are equivalent up to node reordering. On the other hand, augmentations such as cropping may result in qualitatively different super-pixel graphs. Here, it is unclear if the super-pixel graph obtained after augmentation preserves task-relevant information, even if cropping is information preserving with respect to the original image. Therefore, it is not trivial to identify successful augmentations in the abstracted domain that will also be successful in graph space.

Given the difficulty of identifying augmentations that perturb super-pixel graph topology but also preserve task-relevant information, we focus on image space augmentations that lead to modified node features in the super-pixel graph. Specifically, we select *random colorization* as the TAA as it (i) preserves task-relevant information as color is not relevant property when classifying digits, (ii) breaks view symmetry because the node features of augmented samples are different and (iii) introduces a harmless invariance to colorization. We briefly note that augmentations are generally selected to introduce invariances that are useful to the downstream task. For example, cropping results in occlusion invariance, which is useful for classification tasks where objects are often partially covered [52]. Here, we take a complementary approach where augmentations introduce harmless information (color) and the model learns to ignore it. This can be a useful strategy when it is difficult to clearly identify potentially useful invariances for a given task.

Results. In Table 5, we observe that training with an information-preserving, TAA (colorizing) improves accuracy for both SimSiam and SimCLR. While BYOL generally performs worse than SimSiam and SimCLR, colorizing is still within standard deviation of DAGAs. Composing augmentations with colorizing would likely further improve performance, but this investigation is left to future work. This confirms that learning invariance to irrelevant information, as determined by knowledge of the original data modality, is indeed a viable strategy for creating TAAs. Moreover, we note that randomly-initialized models have 37.79% accuracy, indicating that super-pixel data can serve as a sufficiently complex benchmark for future GCL evaluation [14] (see Appendix C for affinity and representational similarity analysis).

5 CONCLUSION

In this work, we discuss limitations in the evaluation and design of existing instance-discrimination GCL frameworks, and introduce new improved practices. In two case studies, we show the benefits of adhering to these practices, particularly the benefits using task-aware augmentations. First, through our analysis, we show that domain-agnostic graph augmentations do not preserve task-relevant information and lead to weakly discriminative representations. We then demonstrate that benchmark graph classification datasets are not appropriate for evaluating GCL frameworks by contextualizing recent theoretical work in VCL. Indeed, we show that the strong inductive bias of randomly initialized, untrained GNNs obfuscates GCL framework inefficiencies. While we acknowledge the community is moving toward larger and more extensive benchmarks [14], we emphasize that it is fundamentally incorrect to continue evaluating GCL on legacy graph classification benchmarks. Furthermore, on two case studies with practically complex tasks, we show how to use domain knowledge to perform information-preserving, task-aware augmentation and achieve significant improvements over training with domain-agnostic graph augmentations. In summary, GCL is an exciting new direction in unsupervised graph representation learning and our work can inform the evaluation of new methods as well as help practitioners design task-aware augmentations.

ACKNOWLEDGMENTS

We thank Jay Thiagarajan and Mark Heimann for helpful discussions on the project. This work is partially supported by the National Science Foundation under CAREER Grant No. IIS 1845491, Army Young Investigator Award No. W9-11NF1810397, and Adobe, Amazon, Facebook, and Google faculty awards. Any opinions, findings, and conclusions or recommendations expressed here are those of the author(s) and do not reflect the views of funding parties.

REFERENCES

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurélien Lucchi, Pascal Fua, and Sabine Süsstrunk. 2012. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 11 (2012), 2274–2282.

[2] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. 2019. A Theoretical Analysis of Contrastive Unsupervised Representation Learning. *arXiv* abs/1902.09229 (2019).

[3] Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Dipendra Misra. 2022. Investigating the Role of Negatives in Contrastive Representation Learning. In *Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*.

[4] Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenbing Huang, Yu Rong, and Junzhou Huang. 2020. Rumor Detection on Social Media with Bi-Directional Graph Convolutional Networks. In *Proc. Association for the Advancement of Artificial Intelligence Conf. on Artificial Intelligence (AAAI)*.

[5] Karsten M. Borgwardt, Cheng Soon Ong, Stefan Schönauer, S. V. N. Vishwanathan, Alexander J. Smola, and Hans-Peter Kriegel. 2005. Protein function prediction via graph kernels. In *Proc. Int. Conf. on Intelligent Systems for Molecular Biology*.

[6] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging Properties in Self-Supervised Vision Transformers. In *Proc. Int. Conference on Computer Vision (ICCV)*.

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[9] Xinlei Chen and Kaiming He. 2021. Exploring Simple Siamese Representation Learning. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[10] Xinlei Chen, Saining Xie, and Kaiming He. 2021. An Empirical Study of Training Self-Supervised Vision Transformers. In *Proc. Int. Conference on Computer Vision (ICCV)*.

[11] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Velickovic. 2020. Principal Neighbourhood Aggregation for Graph Nets. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[12] Yingtong Dou, Kai Shu, Congying Xia, Philip S. Yu, and Lichao Sun. 2021. User Preference-aware Fake News Detection. In *Proc. of the Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*.

[13] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[14] Vijay Prakash Dwivedi, Chaitanya K. Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2020. Benchmarking Graph Neural Networks. *arXiv* abs/2003.00982 (2020).

[15] Yin Fang, Haihong Yang, Xiang Zhuang, Xin Shao, Xiaohui Fan, and Huajun Chen. 2021. Knowledge-aware Contrastive Molecular Graph Learning. *arXiv* (2021).

[16] Matthias Fey and Jan E. Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

[17] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. 2020. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[18] Tao Guo and Baojiang Cui. 2022. Web Page Classification Based on Graph Neural Network. In *Innovative Mobile and Internet Services in Ubiquitous Computing*.

[19] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[20] Yi Han, Shanika Karunasekera, and Christopher Leckie. 2021. Continual Learning for Fake News Detection from Social Media. In *Int. Conf. on Artificial Neural Networks*.

[21] Jeff Z. HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. 2021. Provable Guarantees for Self-Supervised Deep Learning with Spectral Contrastive Loss. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*.

[22] Kaveh Hassani and Amir Hosein Khas Ahmadi. 2020. Contrastive Multi-View Representation Learning on Graphs. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[23] Kaveh Hassani and Amir Hosein Khas Ahmadi. 2022. Learning Graph Augmentations to Learn Graph Representations. *arXiv* (2022). arXiv:2201.09830

[24] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[25] Mark Heimann, Tara Safavi, and Danai Koutra. 2019. Distribution of Node Embeddings as Multiresolution Features for Graphs. In *2019 IEEE International Conference on Data Mining (ICDM)*.

[26] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. 2019. Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*.

[27] R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[28] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay S. Pande, and Jure Leskovec. 2020. Strategies for Pre-training Graph Neural Networks. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[29] Doyeong Hwang, Soojung Yang, Yongchan Kwon, Kyung-Hoon Lee, Grace Lee, Hanseok Jo, Seyeol Yoon, and Seongok Ryu. 2020. Comprehensive Study on Molecular Supervised Learning with Graph Neural Networks. *J. Chem. Inf. Model.* 60, 12 (2020), 5936–5945.

[30] Ashraful Islam, Chun-Fu Chen, Rameswar Panda, Leonid Karlinsky, Richard J. Radke, and Rogério Feris. 2021. A Broad Study on the Transferability of Visual Representations with Contrastive Learning. In *Proc. Int. Conference on Computer Vision (ICCV)*.

[31] Weiwei Jiang and Jiayun Luo. 2021. Graph Neural Network for Traffic Forecasting: A Survey. *ArXiv* abs/2101.11174 (2021).

[32] Zekarias T. Kefato and Sarunas Girdzijauskas. 2021. Self-supervised Graph Neural Networks without explicit negative sampling. *arXiv* abs/2103.14958 (2021).

[33] Zekarias T. Kefato, Sarunas Girdzijauskas, and Hannes Stärk. 2021. Jointly Learnable Data Augmentations for Self-Supervised GNNs. *arXiv* abs/2108.10420 (2021).

[34] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[35] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[36] Boris Knyazev, Graham W. Taylor, and Mohamed R. Amer. 2019. Understanding Attention and Generalization in Graph Neural Networks. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[37] Risi Imre Kondor and John Lafferty. 2002. Diffusion kernels on graphs and other discrete structures. In *In Proceedings of the ICML*.

[38] Kezhi Kong, Guohao Li, Mucong Ding, Zuxuan Wu, Chen Zhu, Bernard Ghanem, Gavin Taylor, and Tom Goldstein. 2020. FLAG: Adversarial Data Augmentation for Graph Neural Networks. *arXiv* abs/2010.09891 (2020).

[39] Nils M. Kriege and Petra Mutzel. 2012. Subgraph Matching Kernels for Attributed Graphs. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[40] Namkyeong Lee, Junseok Lee, and Chanyoung Park. 2021. Augmentation-Free Self-Supervised Learning on Graphs. *arXiv* abs/2112.02472 (2021).

[41] Yixin Liu, Shirui Pan, Ming Jin, Chuan Zhou, Feng Xia, and Philip S. Yu. 2021. Graph Self-Supervised Learning: A Survey. *arXiv* abs/2103.00111 (2021).

[42] Raphael Gontijo Lopes, Sylvia J. Smullin, Ekin D. Cubuk, and Ethan Dyer. 2020. Tradeoffs in Data Augmentation: An Empirical Study. In *Int. Conf. on Learning Representations (ICLR)*.

[43] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[44] Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M. Bronstein. 2019. Fake News Detection on Social Media using Geometric Deep Learning. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

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

[46] Giannis Nikolentzos, Antoine J.-P. Tixier, and Michalis Vazirgiannis. 2020. Message Passing Attention Networks for Document Understanding. In *Association for the Advancement of Artificial Intelligence Conf. on Artificial Intelligence (AAAI)*.

[47] Kento Nozawa and Issei Sato. 2021. Understanding Negative Samples in Instance Discriminative Self-supervised Representation Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[48] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report. Stanford InfoLab.

[49] Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In *Proc. Annual Meeting of the Assoc. for Computational Linguistics (ACL)*.

[50] Hyeonjin Park, Seunghun Lee, Sihyeon Kim, Jinyoung Park, Jisu Jeong, Kyung-Min Kim, Jung-Woo Ha, and Hyunwoo J. Kim. 2021. Metropolis-Hastings Data Augmentation for Graph Neural Networks. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[51] Ben Poole, Sherjil Ozair, Aaron van den Oord, Alexander A. Alemi, and George Tucker. 2019. On Variational Bounds of Mutual Information. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[52] Senthil Purushwalkam and Abhinav Gupta. 2020. Demystifying Contrastive Self-Supervised Learning: Invariances, Augmentations and Dataset Biases. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[53] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In *Proc. of ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining (KDD)*.

[54] Joshua David Robinson, Li Sun, Ke Yu, kayhan Batmanghelich, Stefanie Jegelka, and Suvrit Sra. 2021. Can contrastive learning avoid shortcut solutions?. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[55] Benedek Rozemberczki, Oliver Kiss, and Rik Sarkar. 2020. Karate Club: An API Oriented Open-source Python Framework for Unsupervised Learning on Graphs. In *Proc. of ACM Int. Conf. on Information and Knowledge Management (CIKM)*.

[56] Vadeem Safronov. 2021. Almost Free Inductive Embeddings Out-Perform Trained Graph Neural Networks in Graph Classification in a Range of Benchmarks. https://towardsdatascience.com/almost-free-inductive-embeddings-out-perform-trained-graph-neural-networks-in-graph-classification-651ace368bc1. Accessed: 2021-10-05.

[57] Ramprasaath R. Selvaraju, Karan Desai, Justin Johnson, and Nikhil Naik. 2021. CASTing Your Model: Learning to Localize Improves Self-Supervised Representations. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[58] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. 2011. Weisfeiler-Lehman Graph Kernels. *J. Mach. Learn. Res.* 12 (2011), 2539–2561.

[59] Weijing Shi and Ragunathan (Raj) Rajkumar. 2020. Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[60] Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020. FakeNewsNet: A Data Repository with News Content, Social Context, and Spatiotemporal Information for Studying Fake News on Social Media. In *Big Data*.

[61] Kihyuk Sohn. 2016. Improved Deep Metric Learning with Multi-class N-pair Loss Objective. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*.

[62] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. 2020. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[63] Susheel Suresh, Pan Li, Cong Hao, and Jennifer Neville. 2021. Adversarial Graph Augmentation to Improve Graph Contrastive Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[64] Alex Tamkin, Mike Wu, and Noah D. Goodman. 2021. Viewmaker Networks: Learning Views for Unsupervised Representation Learning. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[65] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Rémi Munos, Petar Velickovic, and Michal Valko. 2021. Bootstrapped Representation Learning on Graphs. *arXiv* (2021).

[66] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Rémi Munos, Petar Velickovic, and Michal Valko. 2021. Bootstrapped Representation Learning on Graphs. *arXiv* abs/2102.06514 (2021).

[67] Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2020. Contrastive Multiview Coding. In *Proc. European Conf. on Computer Vision (ECCV)*.

[68] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. 2020. What Makes for Good Views for Contrastive Learning?. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[69] Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2021. Self-supervised Learning from a Multi-view Perspective. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[70] Michael Tschannen, Josip Djolonga, Paul K. Rubenstein, Sylvain Gelly, and Mario Lucic. 2020. On Mutual Information Maximization for Representation Learning. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[71] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. *arXiv* abs/1807.03748 (2018).

[72] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. 2014. scikit-image: image processing in Python.

[73] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[74] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. 2019. Deep Graph Infomax. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[75] Vikas Verma, Thang Luong, Kenji Kawaguchi, Hieu Pham, and Quoc V. Le. 2021. Towards Domain-Agnostic Contrastive Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[76] Julius von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. 2021. Self-Supervised Learning with Data Augmentations Provably Isolates Content from Style. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[77] Nikil Wale and George Karypis. 2006. Comparison of Descriptor Spaces for Chemical Compound Retrieval and Classification. In *Int. Conf. on Data Mining (ICDM)*.

[78] Jason W. Wei and Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *Proc. Conf. on Empirical Methods in Natural Language Processing and Int. Joint Conf. on Natural Language Processing (EMNLP-IJCNLP)*.

[79] Zixin Wen and Yuanzhi Li. 2021. Towards Understanding the Feature Learning Process of Self-supervised Contrastive Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[80] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. 2018. Unsupervised Feature Learning via Non-Parametric Instance-level Discrimination. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[81] Tete Xiao, Xiaolong Wang, Alexei A. Efros, and Trevor Darrell. 2021. What Should Not Be Contrastive in Contrastive Learning. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[82] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[83] Minghao Xu, Hang Wang, Bingbing Ni, Hongyu Guo, and Jian Tang. 2021. Self-supervised Graph-level Representation Learning with Local and Global Structure. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[84] Yujun Yan, Jiong Zhu, Marlena Duda, Eric Solarz, Chandra Sekhar Sripada, and Danai Koutra. 2019. GroupINN: Grouping-based Interpretable Neural Network for Classification of Limited, Noisy Brain Data. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*.

[85] Pinar Yanardag and S. V. N. Vishwanathan. 2015. Deep Graph Kernels. In *Proc. of ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining (KDD)*.

[86] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. 2021. Graph Contrastive Learning Automated. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[87] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph Contrastive Learning with Augmentations. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[88] Yuning You, Tianlong Chen, Zhangyang Wang, and Yang Shen. 2020. When Does Self-Supervision Help Graph Convolutional Networks?. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[89] Yuning You, Tianlong Chen, Zhangyang Wang, and Yang Shen. 2022. Bringing Your Own View: Graph Contrastive Learning without Prefabricated Data Augmentations. In *Proc. ACM Int. Conf. Web Search and Data Mining (WSDM)*.

[90] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. 2018. Generative Image Inpainting with Contextual Attention. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[91] Zhiping Zeng, Anthony KH Tung, Jianyong Wang, Jianhua Feng, and Lizhu Zhou. 2009. Comparing stars: On approximating graph edit distance. In *Proc. of the Very Large Data Base (VLDB) Endowment*.

[92] Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. 2020. Every Document Owns Its Structure: Inductive Text Classification via Graph Neural Networks. In *Proc. Annual Meeting of the Assoc. for Computational Linguistics (ACL)*.

[93] Tong Zhao, Yozen Liu, Leonardo Neves, Oliver J. Woodford, Meng Jiang, and Neil Shah. 2020. Data Augmentation for Graph Neural Networks. In *Proc. Association for the Advancement of Artificial Intelligence Conf. on Artificial Intelligence (AAAI)*.

[94] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph Contrastive Learning with Adaptive Augmentation. In *Proc. The ACM Web Conf. (WWW)*.

[95] Roland S. Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. 2021. Contrastive Learning Inverts the Data Generating Process. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[96] Marinka Zitnik, Rok Sosič, and Jure Leskovec. 2018. Prioritizing network communities. *Nature Communications* (2018).

A EXPERIMENTAL DETAILS OF SECTION 3

For Secs. 3.2, 3.3 experiments, we use a GIN-based encoder [82] similar to InfoGraph [62] and GraphCL [87] for all datasets but (DEEZER, GOSSIPCOP, GITHUB-SGZR). PNA is used for (DEEZER,GITHUB-SGZR) to stabilize Infograph's loss and in Sec. 3.1. For GOSSIPCOP, the encoder is based off PyG's implementation [16]: 1 GCN Layer, 1 Linear Layer, embedding dimension = 128, Optimizer = Adam [34], LR = 0.001, # of Epochs = 25, batch size = 128.

Sec. 3.1 Experimental Setup: The following training configuration is used: # of Layers = 3, LR = 0.01, # of Epochs = 30, Batch-Size = 32. Models are trained on a Nvidia Tesla K80 GPU with Adam. A batch-norm layer is included between the output of the backbone and cross entropy layer. For augmentations, we follow [87] and stochastically apply node dropping at 20% of graph size and subgraph dropping at 20% of graph size.

Sec. 3.2 Experimental Setup: 3-layer GIN model with hidden dimension, learning rate, and epochs trained of (32, NA, NA) for RAND (Random Initialization), (512,0.001,20) for InfoGraph, and (32,0.01,20) for GraphCL. Adam and Nvidia Tesla K80 GPUs (12-GB GPU) were used to train all models. Results for MVGRL ([22]) are not included as we consistently witnessed Out-Of-Memory errors. Results are reported over 3 seeds. *Additional Results:* Fig. 5a includes additional results for PROTEINS, NCI1 and DD datasets.

Sec. 3.3 Experimental Setup: For all datasets, excluding DEEZER and GITHUB-SGZR, we report results from GraphCL and InfoGraph. We use the same GIN encoder as GraphCL when reporting the performance of randomly initialized models for these datasets. On GITHUB-SGZRS, InfoGraph training time on exceeds eights hours using a NVIDIA Tesla P100. *Additional Results:* See Table 6. We find that the inductive bias of GNNs is strong across different architectures (GraphSAGE, PNA, GCN, and GAT).

Table 6: *Inductive Bias*: Additional results.

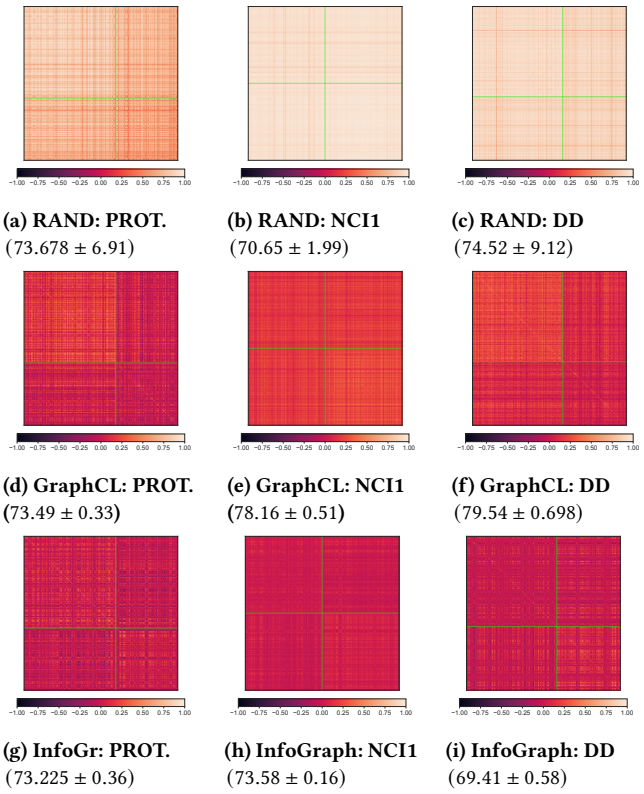
GraphSAGE	3 Layer	4 Layer	5 Layer	GraphCL	InfoGraph
MUTAG	0.85 ± 0.005	0.85 ± 0.006	0.85 ± 0.005	0.82 ± 0.040	0.85 ± 0.005
PROTEINS	0.73 ± 0.004	0.73 ± 0.003	0.74 ± 0.005	0.75 ± 0.002	0.74 ± 0.008
NCI1	0.74 ± 0.003	0.75 ± 0.006	0.73 ± 0.011	0.78 ± 0.000	0.79 ± 0.002
DD	0.77 ± 0.006	0.78 ± 0.002	0.78 ± 0.005	0.80 ± 0.008	0.77 ± 0.010
REDDIT-B	0.85 ± 0.014	0.83 ± 0.016	0.83 ± 0.005	–	0.66 ± 0.137
IMDB-B	0.66 ± 0.012	0.81 ± 0.008	0.81 ± 0.008	–	–
PNA	3 Layer	4 Layer	5 Layer	GraphCL	InfoGraph
MUTAG	0.88 ± 0.011	0.88 ± 0.010	0.89 ± 0.009	0.86 ± 0.023	0.90 ± 0.014
PROTEINS	0.74 ± 0.003	0.74 ± 0.012	0.74 ± 0.005	0.74 ± 0.007	0.74 ± 0.003
NCI1	0.67 ± 0.008	0.68 ± 0.011	0.68 ± 0.010	0.78 ± 0.008	0.77 ± 0.019
DD	0.76 ± 0.014	0.76 ± 0.002	0.76 ± 0.008	0.80 ± 0.008	0.76 ± 0.006
REDDIT-B	0.90 ± 0.003	0.88 ± 0.014	0.89 ± 0.010	0.92 ± 0.006	0.92 ± 0.006
IMDB-B	0.72 ± 0.007	0.68 ± 0.011	0.68 ± 0.010	0.71 ± 0.009	0.71 ± 0.009
GCN	3 Layer	4 Layer	5 Layer	GraphCL	InfoGraph
MUTAG	0.85 ± 0.003	0.85 ± 0.004	0.85 ± 0.005	0.82 ± 0.013	0.85 ± 0.003
PROTEINS	0.74 ± 0.003	0.73 ± 0.007	0.74 ± 0.004	0.75 ± 0.004	0.75 ± 0.003
NCI1	0.76 ± 0.004	0.75 ± 0.001	0.75 ± 0.002	0.78 ± 0.008	0.79 ± 0.007
DD	0.78 ± 0.002	0.77 ± 0.012	0.78 ± 0.003	0.79 ± 0.007	0.76 ± 0.003
REDDIT-B	0.52 ± 0.005	0.51 ± 0.003	0.52 ± 0.005	0.92 ± 0.002	0.80 ± 0.062
IMDB-B	0.54 ± 0.001	0.57 ± 0.016	0.58 ± 0.008	0.71 ± 0.011	0.62 ± 0.070
GAT	3 Layer	4 Layer	5 Layer	GraphCL	InfoGraph
MUTAG	0.84 ± 0.003	0.85 ± 0.009	0.84 ± 0.003	0.81 ± 0.032	0.85 ± 0.013
PROTEINS	0.74 ± 0.002	0.74 ± 0.005	0.74 ± 0.006	0.74 ± 0.007	0.74 ± 0.005
NCI1	0.76 ± 0.009	0.75 ± 0.004	0.76 ± 0.002	0.78 ± 0.004	0.70 ± 0.040
DD	0.78 ± 0.005	0.77 ± 0.006	0.79 ± 0.001	0.79 ± 0.003	0.76 ± 0.005
REDDIT-B	0.52 ± 0.005	0.53 ± 0.004	0.52 ± 0.012	0.75 ± 0.004	–
IMDB-B	0.51 ± 0.004	0.51 ± 0.009	0.50 ± 0.005	0.51 ± 0.007	–

B DOCUMENT CLASSIFICATION

In Sec. 4.1, we demonstrate the benefits of using task-aware augmentations on a graph-based document classification task.

Experimental Setup: We use the model, code base and default settings of [46]. Models are trained using Adam: lr = 0.001, weight-decay = 1e-4 and cosine scheduler (T=8). We use the code (https://github.com/jasonwei20/eda-nlp) and augmentations by [78]. Synonym replacement, random deletion, random insertion and random swapping are applied at 5%, 10%, 5%, 5% of sentence length respectively. We generate an augmented version of each sentence for every training epoch. For domain agnostic augmentations, we apply random node dropping (10%) to generate one view. The other view is generated by applying random node or subgraph dropping (10%).

As noted in Sec. 4.1, natural language augmentations can be directly in graph space. We provide proof of concept using the synonym replacement augmentation. In Table 7, results are reported for a model trained with synonym replacement and graph space equivalent, node replacement at 5%. This model achieves comparable accuracy to the original task-aware augmentations. We suspect that synonym replacement is crucial for this task.



(a) RAND: PROT. (73.678 ± 6.91) **(b) RAND: NCI1** (70.65 ± 1.99) **(c) RAND: DD** (74.52 ± 9.12)

(d) GraphCL: PROT. (73.49 ± 0.33) **(e) GraphCL: NCI1** (78.16 ± 0.51) **(f) GraphCL: DD** (79.54 ± 0.698)

(g) InfoGr: PROT. (73.225 ± 0.36) **(h) InfoGraph: NCI1** (73.58 ± 0.16) **(i) InfoGraph: DD** (69.41 ± 0.58)

Figure 5: *Representational Similarity:* In addition to MUTAG (Figure 2), we provide results on PROTEINS, NCI1 and DD. Random inductive bias is most noticeable on MUTAG and PROTEINS. Note that the intra-class similarity can be low for GraphCL and InfoGraph.

Table 7: *Document Classification*: We use the same augmentations as in Table 4. Text-to-Graph augmentations perform synonym replacement as modifying node features.

Augmentation	(SimSiam) KNN Acc.	(BYOL) KNN Acc.
S. vs S. (ws = 2)	62.62 ± 3.21	66.25 ± 2.65
S. vs N. (ws = 2)	57.35 ± 2.47	62.83 ± 2.82
Text-Space (ws = 2)	**83.69 ± 0.01**	**82.69 ± 1.98**
Text-to-Graph (ws = 2)	**83.33 ± 1.29**	78.16 ± 2.11
S. vs S. (ws = 4)	63.70 ± 8.71	67.53 ± 5.00
S. vs N. (ws = 4)	54.77 ± 1.42	65.99 ± 2.78
Text-Space (ws = 4)	**83.29 ± 0.9**	**72.91 ± 4.97**
Text-to-Graph Space (ws = 4)	**84.67 ± 1.57**	**77.96 ± 2.04**

C SUPER-PIXEL CLASSIFICATION

In Sec. 4.2, we demonstrate the benefits of using task-aware augmentations via a case study on MNIST superpixel classification.

Experimental Setup: 50K images are used for training, 10K for validation, and 10K for testing. We follow the same procedure as [14] to convert images to superpixel graphs: SLIC ([1]) is used to extract superpixels from the image. Then, a kNN graph is constructed between the superpixels. Node features are RGB values and (x, y) coordinates of superpixels. Classification is performed using three CL frameworks: SimSiam ([9]), SimCLR ([8]), and BYOL ([17]). The same hyper-parameters and architecture are used for all frameworks. Specifically, we use a 5-Layer GIN model closely following [14]. This model is converted from DGL (https://www.dgl.ai) to PyG ([16]). The following hyper-parameters are used: LR=5e-4, Hidden-Dim =110, Epochs=80, Batch-size = 128. The Adam ([34]) Optimizer is used for training. The projector is a 2-layer MLP. The predictor is a 2-layer MLP. Predictor hidden dimension is 1028. Bottleneck dimension is 128. Results are reported over 3 seeds. DAGAs are random node dropping (at 20% and 30%). The task-aware augmentation is random colorizing, performed using Scikit-Image ([72]). As discussed in the main text, colorizing can be represented as transformation on node features as well.

Additional Results: [75] proposes to mix-up samples at either the input or hidden representation level as an alternative to domain-specific augmentations. However, we find that [75] under-performs both node-dropping and colorizing, despite tuning the mixing parameter, α (see Table. 8).This indicates that context-aware and topological augmentations are still important to GCL. Table 9 shows intra/inter similarity and Table 10 shows the affinity.

Table 8: *Comparison to [75].* Results only reported for SimCLR, as it performs better than SimSiam and BYOL in preceding experiments.

Rand Init.	ND (20%)	ND (30%)	Colorize	DACL [75]
37.79 ± 0.03	68.56 ± 0.16	68.07 ± 0.37	**73.67 ± 0.10**	59.94 ± 0.01

Table 9: *Super-pixel, Rep. Similarity.* Avg. intraclass and interclass cosine similarity is reported. Colorizing produces representations with the largest difference between intra- vs. inter- class similarity, indicating that representations are well-separated.

Method	Aug.	Intra. Sim	Inter Sim.	Abs. Diff	Rel. Diff	Acc.
SimCLR	ND (20%)	86.671	78.622	8.04	0.0928	68.56 ± 0.16
SimCLR	ND (30%)	87.03	79.05	7.987	0.091	68.07 ± 0.37
SimCLR	Colorizing	80.801	67.812	12.988	0.1607	**73.67 ± 0.10**

Table 10: *Super-pixel Affinity.* Supervised, clean train accuracy is 90.01% and clean test accuracy is 88.69%.

Aug.	Aug. Train Acc.	Aug. Test Acc.
ND (20%)	39.42 ± 0.011	40.29 ± 0.054
ND (30%)	29.19 ± 0.01	29.09 ± 0.036
Colorizing	**47.86 ± 0.05**	**48.97 ± 0.03**

D ADDITIONAL RELATED WORK

Graph Data Augmentation. [93] train a neural edge predictor to increase homophily by adding edges between nodes expected to be of the same class and break edges between nodes of expected dissimilar classes. However, this approach is expensive and not applicable to graph classification. [38] focus on feature augmentations because it is easier than designing information preserving topological transformations. They add adversarial perturbations to node features as augmentations. In unsupervised settings, labels are not available and cannot be used for the adversarial perturbation, so the proposed approach is not directly applicable. Since the writing of this paper, several recent works have been proposed that perform automatic data-augmentation, some of which we briefly describe in Table 11.

Graph Self-Supervised Learning. Several paradigms for self-supervised learning in graphs have been recently explored, including the use of pre-text tasks, multi-tasks, and unsupervised learning. See [41] for an up-to-date survey. Graph pre-text tasks are often reminiscent of image in-painting tasks [90], and seek to complete masked graphs and/or node features ([28, 88]). Other successful approaches include predicting graph level or property level properties during pre-training or part of regular training to prevent overfitting ([28]). These tasks often must be carefully selected to avoid negative transfer between tasks. Many unsupervised approaches have also been proposed. [62, 74] draw inspiration from [27] and maximize the mutual information between global and local representations; MVGRL ([22]) contrasts different views at multiple granularities similar to [71]; [32, 53, 66, 87, 94] use augmentations to generate views for contrastive learning. See Table 11 for a summary of the augmentations used.

Table 11: *Selected GCL Frameworks*

Method	Augmentations
BGRL [66]	Edge Dropping, Attr. Masking
GCA [94]	Edge Dropping, Attr. Masking (both weighted by centrality)
GCC [53]	RWR Subgraph Extraction of Ego Network
GraphCL [87]	Node Dropping, Edge Adding/Dropping, Attr. Masking, Subgraph Extraction
MVGRL [22]	PPR Diffusion + Sampling
SelfGNN [32]	Attr. Splitting, Attr. Standardization + Scaling, Local Degree Profile, Paste + Local Degree Profile
JOAO [86]	Min-Max Optimization to adaptively and dynamically select from DAGA set
GraphSurgeon [33]	Learnable Feature Augmentors that can be applied pre/post encoding
BYOV [89]	Uses graph generation (regularized by InfoMin + InfoBottleNeck) as viewmaker
AdvGCL [63]	Adversarial/MinMax Optimization over learnable augmentations
AF-GRL [40]	Finds node-level positive samples sharing "local structure and global semantics"
LG2AR [23]	Learns a policy over augmentations and their respective strengths without bi-level optimization