A Content-First Benchmark for Self-Supervised Graph Representation Learning

Puja Trivedi
University of Michigan

Mark Heimann
Lawrence Livermore Natl. Laboratory

Ekdeep Singh Lubana
University of Michigan

Danai Koutra
University of Michigan

Jayaraman J. Thiagarajan
Lawrence Livermore Natl. Laboratory

ABSTRACT

Current advances in unsupervised representation learning (URL) have primarily been driven by novel contrastive learning and reconstruction based paradigms. Recent work finds the following properties to be critical for *visual* URL's success: *invariance* to task-irrelevant attributes, *recoverability* of labels from augmented samples, and *separablility* of classes in some latent space. However, these properties are hard to measure or sometimes unsupported when using commonly adopted *graph* augmentations and benchmarks, making it difficult to evaluate the merits of different URL paradigms or augmentation strategies. For example, on several benchmark datasets, we find that popularly used, generic graph augmentations (GGA) do not induce task-relevant invariance. Moreover, GGA's recoverability cannot be directly evaluated as it is unclear how graph semantics, potentially altered by augmentation, are related to the task. Through this work, we introduce a synthetic data generation process that allows us to control the amount of task-irrelevant (style) and task-relevant (content) information in graph datasets. This construction enables us to define oracle augmentations that induce task-relevant invariances and are recoverable by design. The class separability, i.e., hardness of a task, can also be altered by controlling the degree of irrelevant information. Our proposed process allows us to evaluate how varying levels of style affects the performance of graph URL algorithms and augmentation strategies. Overall, this data generation process is valuable to the community for better understanding limitations of proposed graph URL paradigms that are otherwise not apparent through standard benchmark evaluation.

1 INTRODUCTION

For many practical machine learning tasks, labeled data is expensive [66], difficult to obtain [48, 49], and potentially biased [5, 34]. In such scenarios, unsupervised representation learning (URL) offers an alternative paradigm that not only enables the use of larger, unlabeled datasets [55] but has also been shown to produce more robust [20, 30], transferable [12, 22] and semantically consistent [7] representations. Recent success of visual URL has primarily been driven by novel contrastive learning (CL) [6–8, 10, 19, 21, 40, 44, 62] as well as improved reconstruction-based algorithms [13, 18]; many of which have direct, graph analogues [17, 37, 39, 59, 64].

The empirical success of visual CL has lead to a surge of efforts seeking to gain insights into its behavior [1, 14, 16, 32, 41, 47, 51, 65]. Underlying many of these analyses is the assumption that there exists a latent space that satisfies the following properties [2, 52]: (i) labels of augmented samples are generally *recoverable* from the natural sample from which they were generated; and (ii) samples (and corresponding augmentations) from different underlying classes are *separable* in this latent space. Moreover, works studying view generation for CL have found that augmentations should generate views that only share the minimum information relevant for downstream tasks [41] and introduce useful, task-dependent *invariances* [32, 43]. Due to the continuous representation of natural images and well-designed augmentation strategies, Wei et al. empirically show these assumptions are indeed satisfied by vision URL methods [52].

While CL has become increasingly popular for graph URL, it remains unclear if the above properties are supported for non-Euclidean, discrete data, and how violating these assumptions may impact the behavior of graph URL. Indeed, graph data augmentation design [28, 38, 58] remains an open research area because it is difficult to determine *prima facie* what changes to a graph's topology or node features will preserve semantics and what invariances might be relevant to the downstream task. While it is possible that *recoverability* can be inherently more difficult to satisfy on graph datasets, the *separability* assumption could also be violated as intermediate points in such a latent space are meaningless in the discrete, structured input space. Therefore, in this paper, we take a data-centric perspective that seeks to understand these properties in the context of graph URL.

Proposed Work. We first show that commonly used generic graph augmentations (GGA) and benchmarks are insufficient for understanding how the above properties affect the performance of graph URL methods and augmentation strategies. Therefore, we introduce a synthetic data generation process that allows for control over the amount of task-irrelevant (style) and task-relevant (content) information in each sample. By controlling the amount of style and content, we are able to vary class separability as well as define augmentations which induce task-relevant invariances and are recoverable by design. Using the proposed process, we evaluate how different graph URL paradigms perform when varying class separability and augmentations' recoverability. Notably, we properly demonstrate that training with task-relevant augmentations is *necessary* for models to perform well across different style vs. content ratios at test time. Overall, the proposed data generation process helps contextualize properties fundamental to the success of visual URL with respect to graph URL paradigms and data augmentation. Our contributions are summarized as follows:

- **Limitations of existing datasets.** On standard benchmarks, we show that despite improved invariance, models trained with GGA have marginal improvements in accuracy compared to untrained encoders. This indicates GGA barely encode task-semantics.
- **Synthetic Data Generation Process.** We propose a synthetic data generation process that allows for control over the amount of

the task irrelevant vs. relevant information in each sample. Using the proposed process, we evaluate graph URL paradigms under varying class separability and augmentation recoverability.

2 BACKGROUND

In this section, we briefly introduce contrastive learning frameworks and properties of useful data augmentation.

Recent advancements in URL have been driven by the CL paradigm, where representations are learned by enforcing representational similarity between positive views of a sample (i.e., augmentations) and dissimilarity between negative views (i.e., different samples). Existing CL frameworks can be broadly categorized based on the mechanism adopted for enforcing this consistency: discriminative frameworks [8, 38, 40, 44, 58, 59] use the InfoNCE loss; approaches that rely only on positive pairs either use Siamese architectures [10, 14, 39] with stop gradient or asymmetric branches, or enforce cluster-level consistency [6, 7] to eliminate the need for negative samples; approaches such as [3, 62] propose to directly reduce redundancy between views. Despite these differences, all methods rely upon aggressive, task-relevant data augmentation strategies to generate views.

Critically, these views are assumed to approximately preserve task-relevant information [43] or equivalently, that labels of views should be recoverable from the underlying sample [16]. Tian et al. further argued that views should *only* share the minimum amount of task-relevant information to perform well on a downstream task, while also achieving invariance to nuisance or irrelevant information. Empirically, Purushwalkam and Gupta also find that data augmentations should induce view invariances that are well-aligned with characteristics of the downstream task. In this work, we focus on commonly used generic graph augmentations (GGA) [59]. GGA are simple feature and topological perturbations, e.g., node dropping, subgraph sampling, edge perturbation and feature masking, that are limited to altering only a fraction of the original sample. While it is difficult to verify the recoverability assumption on benchmark datasets, in Sec. 4.2, we use the proposed data generation process to better understand how augmentation recoverability affects graph URL performance.

3 WHAT BENCHMARKS CANNOT TELL US

An assumption underlying many graph data augmentation techniques is that topological or feature perturbations constrained to a small fraction of the original graph do not alter task-semantics, and hence learning invariance to such perturbations is beneficial to the downstream task. On real-world benchmark datasets, we cannot directly evaluate whether such augmentations are *recoverable* as there is no mechanism to determine the correctness. Therefore, we instead ask if such augmentations induce invariance that is useful to downstream tasks by conducting the following experiment.

Experiment Set-up: We use the following representative graph URL algorithms: (i) *GraphCL* [59], a popular and effective graph CL method; (ii) *GAE*, Graph Autoencoder [26] that uses a reconstruction cost to learn representations; (iii) Augmentation-Augmented Autoencoder [13], which we adapt to graphs to create the Augmentation Augmented Graph Autoencoder (*AAGAE*) that minimizes the reconstruction error between the reconstruction for an augmented sample and the original.; (iv) *SpecCL*, which uses the SpecLoss [16] for contrastive training; (v) *Untrained representations*, which have been observed to be surprisingly competitive baselines [27, 38, 42, 54]. To the best of our knowledge, ours is the first work to evaluate AAGAE and SpecCL for graph URL. We use the same augmentations and encoder architecture as GraphCL. We add a straight-through estimator [23] to GAE/AAGAE's decoder for better training. For more experimental details, including the performance of all methods, please see appendix A.

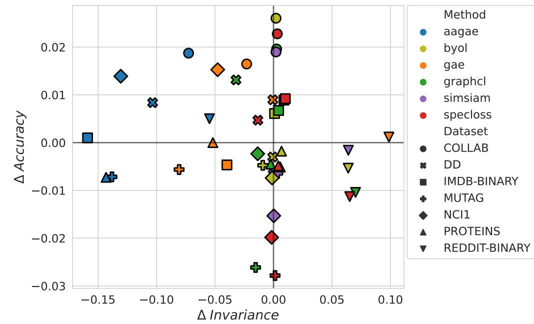

Figure 1: *Invariance vs. KNN Acc.* The change in invariance (Inv.) and accuracy w.r.t. to an untrained encoder is plotted, where Inv. is measured according to [51]. Noticeably, Inv. has not significantly increased for many datasets/methods, improved Inv. does not necessarily entail better performance (see Reddit), and AAGAE/GAE often sees decreased Inv., which we suspect is due to using a decoder.

Training, invariance, and what standard benchmarks cannot tell us. Wang and Isola recently demonstrated that training with InfoNCE is equivalent to optimizing two different properties: *alignment*, or the similarity of positive samples, and *uniformity*, or how well representations are distributed on a hypersphere. Versions of these properties commonly occur in generalization bounds for URL techniques [16, 52]. Intuitively, enforcing positive views to have similar representations, models are expected to become invariant to the given augmentation. To determine if GGA introduces task-relevant invariance (inv.) on benchmark datasets, we compute alignment, defined as the average distance between normalized positive pairs, and kNN accuracy for trained and untrained models. In Fig. 1, we plot the difference in inv. and accuracy, averaged over 10 seeds. We see that many models do not have noticeably better inv. or accuracy with respect to the untrained baseline. Notably, on the Reddit dataset, all methods have improved inv. but do not have significantly better kNN accuracy. In contrast, AAGAE and GAE have *less* inv. than untrained models but improved accuracy; we suspect that the decreased inv. is due to the use of a decoder. Overall, this experiment demonstrates that learning invariance to GGAs is both difficult and often unrelated to task performance. Moreover, given that GGAs have unknown recoverability on standard datasets, and that URL was not able to sufficiently out perform untrained baselines, there is need for new datasets and augmentations where we can better understand the merits of different graph URL paradigms.

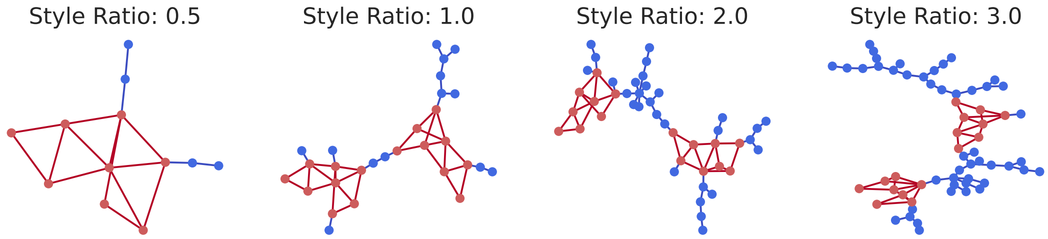

Figure 2: *Synthetic Dataset Generation.* A class-specific motif (shown in red) completely determines the label, and is therefore considered "content". To vary the amount of style, the size of the background tree graph (shown in blue) is a ratio of the number of "content" nodes. Our dataset goes beyond binary, benchmark datasets and allows for content-aware augmentations, a critical component to understanding unsupervised graph representation learning.

4 WHAT SYNTHETIC DATA CAN TELL US

In this section, we first introduce the proposed synthetic data generation process. We then use the proposed process to show how invariance and class separability must be jointly considered when designing augmentations.

4.1 Synthetic Data Generation Process

Given that standard benchmark datasets and augmentation practices are uninformative when evaluating the recoverability and invariance of augmentations, we propose a synthetic data generation process that allows us to understand how the data-dependent assumptions of URL hold for graph datasets. This process not only enables oracle augmentations where recoverability is known, but also allows us some control over dataset separability.

The design of our data generation process is motivated by a recent theoretical work that seeks to understand how CL, data augmentation, and data generation processes are related. Using a latent variable model, von Kügelgen et al. show that self-supervised training with data augmentation is able to recover a *style* vs. *content* partition in the latent representation space. Here, *style* represents information that is irrelevant to the downstream task and can be perturbed (i.e., augmented) without changing sample semantics, while *content* represents task-relevant information and should be preserved. The proposed data generation process creates graph samples that can be decomposed into style vs. content and allows for control over this trade-off (see Figure 2). In doing so, oracle, content-aware augmentations (CAA), with high recoverability, can be evaluated at varying levels of separability, approximated through different style levels.

Generation Process: The proposed data generation process has three components: a set of C motifs, $\mathcal{M}$, that uniquely determine C classes; a random graph generator, $RBG(n)$, parameterized by the number of nodes (we can equivalently define this based on number of edges); and κ, the style multiplier, which controls how much irrelevant information a sample contains. To generate a sample, we attach a randomly generated background graph (*i.e.*, style component) to a motif (*i.e.*, content) according to the style multiplier. This simple process addresses several limitations often encountered in GCL evaluation. Specifically, it (i) allows for varying levels of content-aware augmentation (*i.e.*, edges that can be perturbed in the background graph without harming the motif); (ii) is easily extended beyond binary classification; (iii) contains relatively large number of samples; and (iv) offers a natural test bed for GNN size generalization or transfer learning [57].

4.2 Balancing Style vs. Content

Several real graph datasets can be understood through a style vs. content perspective. For example, in drug discovery tasks [66], molecules can be split into functional groups (content) and carbon rings or scaffold structure (style). One may thus ask: how does varying levels of style vs. content affect the performance of graph URL algorithms, and how do different algorithms benefit from the use of content-aware augmentations? To answer these questions, we conduct the following experiment:

Experiment Details: Let $C = 6$, $\kappa = 4$ and define $RBG(n)$ through a random tree generator, where n is number of the nodes belonging the motif, scaled by κ. Node features are a constant 10-dimensional vector. To increase task difficulty, we randomly insert between 1-3 motif copies into each sample. Using the specified instaniation of the generation process, we train GraphCL, AAGAE, GAE, and SpecLoss with *content-preserving* edge dropping and random edge dropping at 20% and 60% augmentation strength. We also evaluate two recently proposed automated augmentation methods, JOAO [58] and AD-GCL[38]. JOAO is trained with a GGA prior and an expanded GGA prior that includes content-preserving edge dropping. AD-GCL is trained using a learnable edge-dropping augmentor. A 5-layer GIN encoder is used and models are trained for 60 epochs using Adam (lr=0.01). After training, all models are evaluated using the linear probe protocol [8] at varying style ratios. Given that style information is not relevant to the downstream task, we expect models that have truly learned invariance to this information will retain strong performance across different ratios. See appendix A for more model details.

Results. From our results in Figure 3, we make the following observations. First, for reconstruction tasks, as the amount of style increases, the problem inherently becomes harder, as the model must learn to reconstruct increasing amounts of irrelevant information. Indeed, we see that the performance of all reconstruction-based methods decreases as style increases. While content aware augmentations do improve the performance of AAGAE over baseline augmentations, it is unable to match that of contrastive methods. However, with mild random augmentations, AAGAE performs comparably to GAE, and with aggressive random augmentation it performs worse. Second, our analysis suggests that other framework

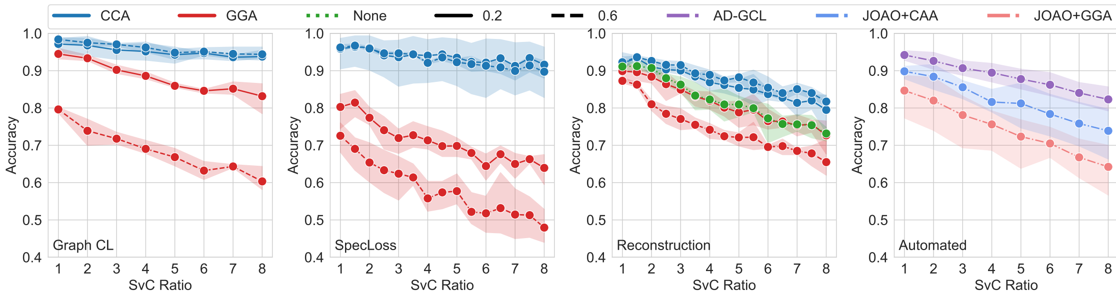

Figure 3: *Style Invariance Across Paradigms:* We evaluate the performance of contrastive and reconstruction approaches with CAAs and GGAs with varying style vs. content ratios. As expected, reconstruction methods perform best in low style regimes, and CAAs improve graph CL performance. Notably, AD-GCL and JOAO do not learn augmentations that induce style-invariance. JOAO is unable to find such a solution even when the prior augmentation set is expanded to include CAAs.

components, such as more expressive architectures [9, 11, 15, 45, 53] and sampling strategies [10, 14, 24], must also be developed before reconstruction-based methods are able to see similar success to visual URL and GCL. Furthermore, we note that the gain from CAA in high-style regimes is much less pronounced for reconstruction approaches than for GCL. This may partially be attributed to increased difficulty in reconstructing larger graphs. More sophisticated decoders and algorithms may help improve performance. Notably, in **??**, we see that automated methods are unable to learn augmentation strategies that induce style invariance. *Indeed, JOAO is unable to find such a solution even when the augmentation prior includes CAAs.* We suspect this is due to their use of bi-level optimization objectives, which are known to be difficult to optimize and prone to finding locally optimal solutions. Overall, this experiment demonstrates that automated methods can be brittle and the proposed benchmark is valuable in evaluation such methods.

4.3 Invariance vs. Separability

We now use our synthetic benchmark to investigate how invariance balances off with the critical assumption of class separability in the latent space. Invariance, while desirable as discussed previously, if considered in isolation could be trivially satisfied by representation collapse, where all graphs are mapped to highly similar representations and are not meaningful for distinguishing classes.

Experimental Setup: Using a synthetic dataset at $\kappa = 6$, we train GraphCL with *content-preserving* edge dropping and random edge dropping at 20% augmentation strength. We compute an invariance score for each natural sample by computing the average cosine similarity of its representation with that 30 different augmented versions. We compute a separability score by dividing the maximum cosine similarity to a sample of the same class by the maximum cosine similarity to a sample of another class.

Generic graph augmentations trade off separability for invariance by collapsing representations. Figure 4 shows kernel density estimates of the number of samples that have a given invariance versus separability, using both random and content-preserving augmentations. Representations from the model trained using GGA have somewhat higher invariance but much lower separability scores. This is likely evidence of model collapse; indeed, with a higher augmentation strength of 60%, we found that using GGA produced invariance and separability scores very close to 1 for all samples, indicating that all samples had similar representations (i.e. strong model collapse). Content-preserving augmentations help GraphCL achieve over an order of magnitude higher separability while still preserving comparably high invariance. We observed similar trends for SpecLoss.

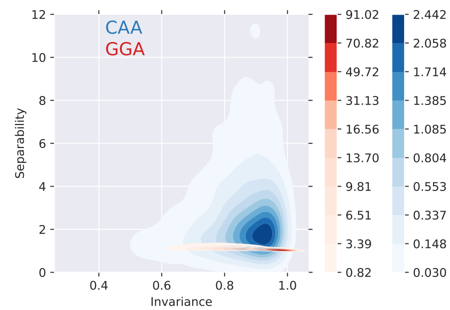

Figure 4: *Inv. vs. Separability.* With $\kappa = 6$ and 20% augmentation strength, GraphCL trained with GGAs produces representations with high inv. but low separability, indicating model collapse. In contrast, using CAAs lead to almost as high inv. but much greater separability.

5 CONCLUSION

In this work, we study the interplay between data-dependent properties, such as recoverability of augmentations and class separability, and the efficacy of graph URL approaches. We first demonstrate that popular, generic graph augmentations do not induce invariance that is useful to downstream tasks. To better understand the benefits of recoverable, i.e. content-aware augmentations, we introduce a systematic synthetic data generation process based on a style-vs-content decomposition. Our work offers an empirical framework to develop graph URL algorithms that are better aligned with data-dependent properties. While simple, the proposed generation process can be extended in several interesting ways. For example, the irrelevant information can be defined using different types of background graph instead of by the ratio. Explainability methods can be used to verify that methods are learning to attend to known, content information, and spectral graph theory can be used to define style vs. content rigorously.

REFERENCES

[1] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. 2019. A Theoretical Analysis of Contrastive Unsupervised Representation Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[2] Maria-Florina Balcan, Avrim Blum, and Ke Yang. 2004. Co-Training and Expansion: Towards Bridging Theory and Practice. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[3] Adrien Bardes, Jean Ponce, and Yann LeCun. 2022. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[4] 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 *Proceedings Thirteenth International Conference on Intelligent Systems for Molecular Biology*.

[5] Joy Buolamwini and Timnit Gebru. 2018. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In *FAT*.

[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 *Proceedings of the International 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] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. Big Self-Supervised Models are Strong Semi-Supervised Learners. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[10] Xinlei Chen and Kaiming He. 2020. Exploring Simple Siamese Representation Learning. *CoRR* abs/2011.10566 (2020).

[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] Linus Ericsson, Henry Gouk, and Timothy M. Hospedales. 2021. How Well Do Self-Supervised Models Transfer?. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[13] William Falcon, Ananya Harsh Jha, Teddy Koker, and Kyunghyun Cho. 2021. AAVAE: Augmentation-Augmented Variational Autoencoders. *CoRR* (2021). arXiv:2107.12329

[14] 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)*.

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

[16] 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)*.

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

[18] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. 2021. Masked Autoencoders Are Scalable Vision Learners. *CoRR* (2021). arXiv:2111.06377

[19] 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)*.

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

[21] 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)*.

[22] 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)*.

[23] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical Reparameterization with Gumbel-Softmax. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[24] Y. Kalantidis, M. Bülent Sariyildiz, N. Pion, P. Weinzaepfel, and D. Larlus. 2020. Hard Negative Mixing for Contrastive Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[25] Zekarias T. Kefato and Sarunas Girdzijauskas. 2021. Self-supervised Graph Neural Networks without explicit negative sampling. *CoRR* abs/2103.14958 (2021). arXiv:210.14958

[26] Thomas N. Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. *CoRR* abs/1611.07308 (2016). arXiv:1611.07308

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

[28] Kezhi Kong, Guohao Li, Mucong Ding, Zuxuan Wu, Chen Zhu, Bernard Ghanem, Gavin Taylor, and Tom Goldstein. 2022. FLAG: Adversarial Data Augmentation for Graph Neural Networks. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*.

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

[30] Hong Liu, Jeff Z. HaoChen, Adrien Gaidon, and Tengyu Ma. 2021. Self-supervised Learning is More Robust to Dataset Imbalance. *CoRR* abs/2110.05025 (2021). arXiv:2110.05025

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

[32] 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)*.

[33] 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 *SIGKDD*. ACM.

[34] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. 2019. Do ImageNet Classifiers Generalize to ImageNet?. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[35] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free!. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

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

[37] 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)*.

[38] 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)*.

[39] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Rémi Munos, Petar Velickovic, and Michal Valko. 2022. Bootstrapped Representation Learning on Graphs.

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

[41] 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)*.

[42] 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 *Proc. of The Web Conf. (WWW)*.

[43] 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)*.

[44] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. *CoRR* (2018).

[45] 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)*.

[46] 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)*.

[47] 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)*.

[48] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. 2016. Anticipating Visual Representations from Unlabeled Video. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[49] Carl Vondrick, Abhinav Shrivastava, Alireza Fathi, Sergio Guadarrama, and Kevin Murphy. 2018. Tracking Emerges by Colorizing Videos. In *Proc. European Conf. on Computer Vision (ECCV)*.

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

[51] Tongzhou Wang and Phillip Isola. 2020. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. In *Proc.*

Int. Conf. on Machine Learning (ICML).

[52] Colin Wei, Kendrick Shen, Yining Chen, and Tengyu Ma. 2021. Theoretical Analysis of Self-Training with Deep Networks on Unlabeled Data. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

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

[54] 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)*.

[55] I. Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. 2019. Billion-scale semi-supervised learning for image classification. *CoRR* abs/1905.00546 (2019). arXiv:1905.00546

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

[57] Gilad Yehudai, Ethan Fetaya, Eli Meirom, Gal Chechik, and Haggai Maron. 2021. From local structures to size generalization in graph neural networks. In *Proc. Int. Conf. on Machine Learning (ICML)*.

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

[59] 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)*.

[60] 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)*.

[61] 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)*.

[62] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. 2021. Barlow Twins: Self-Supervised Learning via Redundancy Reduction. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[63] 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)*.

[64] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Graph Contrastive Learning with Adaptive Augmentation. In *Proc. The Web Conf. (WWW)*.

[65] 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)*.

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

A DATASET GENERATION AND EXPERIMENTAL DETAILS

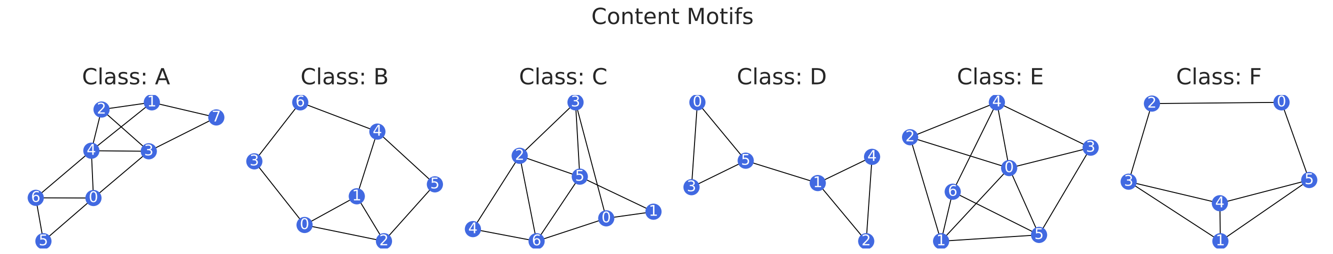

Figure 5: Motifs used to determine class labels.

We use the motifs shown in Fig. A to define a 6 class graph classification task. It is important to ensure that the motifs are not isomorphic, as many GNNs are less expressive than the 1-Weisfeiler Lehman's test for isomorphism ([53]). For each class, 1000 random samples are generated as follows: (i) We randomly select between 1-3 motifs to be in each sample. At this time, motifs all belong to the same class, though this condition could easily be changed for a more difficult task. (ii) We define the number of content nodes, C_n, as the size of the selected motif, scaled by the number of motifs in the sample. (iii) For a given style ratio, we determine the number of possible style nodes as $S_n = \rho C_n$ (iv). We define $RBG(n)$ using networkx's [1] random tree generator: `networkx.generators.trees.random_tree`. We note that other random graph generators would also be well suited for this task. (v) For additional randomness, we create background graphs using $S_n \pm 2$, and also randomly perturb up-to 10% of edges in sample. We repeat this set-up with $\rho \in \{0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 6.5, 7.5, 8.0\}$ to generate the datasets used in Sec **??**.

Experimental Set-up: We follow You et al. for TUDataset experiments. For synthetic datasets we use the following setup. Our encoder is a 5-layer GIN model with mean pooling. We set input node features to be a constant 10-dimensional feature vector, and a hidden layer dimension is 32; we concatenate hidden representations for a representation dimension of 160. Models are pretrained for 60 epochs. Subsequently, we use a linear evaluation protocol and train a linear head for 200 epochs. All models are trained with Adam, lr = 0.01. We use the provided code when training models with JOAO[58] and AD-GCL[38]. We perform an informal grid-search over $\gamma \in \{0.01, 0.1, 1.0\}$ for JOAO and use $\gamma = 0.1$.

Table 1: Benchmarking Graph URL Inductive Bias on Benchmark Datasets. We report the performance of [59], GAE [26], AAGAE and SpecLoss against *un*trained N-layer GIN encoders. Results for GraphCL are taken from the paper, while untrained model results are from [42]. We use the same evaluation protocol and encoder architecture as [59] for trained models. Best results are indicated in bold; results within standard deviation are underlined.

Dataset	Untrained (3)	Untrained(4)	Untrained (5)	GraphCL	GAE	AAGAE	SpecCL
MUTAG (188)	85.76 ± 7.38	86.36 ± 6.51	86.73 ± 10.33	86.80 ± 1.34	87.76 ± 3.00	**88.23 ± 0.98**	86.17 ± 4.11
PROTEINS (1113)	73.64 ± 5.464	74.46 ± 4.09	74.22 ± 2.85	74.39 ± 0.45	**75.36 ± 0.4**	74.77 ± 0.43	74.00 ± 1.58
NCI1 (4110)	70.65 ± 1.99	70.36 ± 3.11	70.49 ± 2.42	77.81 ± 0.41	79.48 ± 0.44	**79.75 ± 1.25**	76.66 ± 0.029
DD (1187)	73.23 ± 8.25	72.15 ± 7.25	77.08 ± 4.18	**78.62 ± 0.40**	78.24 ± 0.67	77.59 ± 0.64	78.43 ± 1.18
RDT-B (2000)	72.34 ± 6.64	64.57 ± 8.03	67.32 ± 7.41	**89.53 ± 0.84**	79.75 ± 1.25	79.95 ± 4.39	79.28 ± 1.049
IMDB-B (1000)	67.22 ± 7.77	61.26 ± 7.01	60.43 ± 5.92	71.14 ± 0.44	**71.70 ± 0.36**	71.26 ± 0.305	71.4 ± 2.19

B RELATED WORK

Graph Data Augmentation: Unlike images, graphs are discrete objects that do not naturally lie in Euclidean space, making it difficult to define meaningful augmentations. Furthermore, while for images or natural language, there may be an intuitive understanding of what changes will preserve task-relevant information, this is not the case for graphs. Indeed, a single edge change can completely change the properties of a molecular graph. Therefore, only a few works consider graph data augmentation. [63] note that a node classification task can be perfectly solved if edges only exist between same class samples. They increase homophily by adding edges between nodes that a neural network predicts belong to the same class and breaking edges between nodes of predicted dissimilar classes. However, this approach is expensive and not applicable to graph classification. [28] argue that information preserving topological transformations are difficult for the aforementioned reasons and instead focus on feature augmentations. Throughout training, they add an adversarial perturbation to node features to improve generalization, computing the gradient of the model weights while computing the gradients of the adversarial

[1] https://networkx.org/documentation/stable/

Table 2: *Dataset Description*

Name	Graphs	Classes	Avg. Nodes	Avg. Edges	Domain
IMDB-BINARY [56]	1000	2	19.77	96.53	Social
REDDIT-BINARY [56]	2000	2	429.63	497.75	Social
MUTAG [29]	188	2	17.93	19.79	Molecule
PROTEINS [4]	1113	2	39.06	72.82	Bioinf.
DD [36]	1178	2	284.32	715.66	Bioinf.
NCI1 [50]	4110	2	29.87	32.30	Molecule

Table 3: Selected Graph Contrastive Learning Frameworks. We provide a brief description of augmentations used by selected frameworks. Most frameworks use random corruptive, sampling, or diffusion-based approaches to generate augmentations.

Method	Augmentations
GraphCL ([59])	Node Dropping, Edge Adding/Dropping, Attribute Masking, Subgraph Extraction
GCC ([33])	RWR Subgraph Extraction of Ego Network
MVGRL ([17])	PPR Diffusion + Sampling
GCA ([64])	Edge Dropping, Attribute Masking (both weighted by centrality)
BGRL ([39])	Edge Dropping, Attribute Masking
SelfGNN ([25])	Attribute Splitting, Attribute Standardization + Scaling, Local Degree Profile, Paste + Local Degree Profile

perturbation to avoid more expensive adversarial training [35]. This approach is not directly applicable to contrastive learning, where label information cannot be used to generate the adversarial perturbation.

Graph Self-Supervised Learning: In graphs, recent works have explored several paradigms for self-supervised learning: see [31] for an up-to-date survey. Graph pre-text tasks are often reminiscent of image in-painting tasks [61], and seek to complete masked graphs and/or node features ([22, 60]). Other successful approaches include predicting auxiliary properties of nodes or entire graphs during pre-training or part of regular training to prevent overfitting ([22]). These tasks often must be carefully selected to avoid negative transfer between tasks. Many contrast-based unsupervised approaches have also been proposed, often inspired by techniques designed for non-graph data. [37, 46] draw inspiration from [21] and maximize the mutual information between global and local representations. MVGRL ([17]) contrasts different views at multiple granularities similar to [44]. [25, 33, 39, 59, 64] use augmentations (which we summarize in Table B) to generate views for contrastive learning. We note that random corruption, sampling or diffusion based approaches used to create generic graph augmentations often do not preserve task-relevant information or introduce meaningful invariances.