Scaffold-based Self-Supervised Learning for Molecular Graphs

Ziyi Liu
Department of Computer Science and Technology
University of Cambridge
zl413@cam.ac.uk

Abstract

Data augmentation has proven a potent approach for many self-supervising contrastive learning methods, where true data can be slightly modified differently into input pairs, such that the model to pre-train can still recognize the original instance after training by matching the representation outputs, making the model robust. Although such methods have also been applied to graphs with GNN methods, applications on molecular graphs in particular requires extra considerations since randomly masking atoms and subgraphs can significantly change the molecular features and will invalidate the augmentation approach. Therefore, we want to develop a tailored approach for contrastive learning on molecules, where scaffolds - key structures in molecules that cannot be altered without changing the molecular properties significantly - are kept intact while we tweak the appended structures onto them. We will use two main methods for constructing such samples: 1) pairing molecules with identical scaffolds as positive samples and 2) augmenting each scaffold into a pair of different molecules of the same scaffold backbone. We then integrate the augmentation precedures with the GNN model pretraining and use them for a variety of downstream tasks with different datasets (BBBP, Freesolv, etc.) for both classification and regression, benchmarked against random-augmentation baselines. We will also benchmark various GNNs' performances against other non-graph-based models (e.g. SVM, Random Forest) to demonstrate their capability in leveraging the scaffolds.

1 Introduction

Encoding molecular information into computationally effective and convenient representations has seen a boost of interests and improvement since the introduction of GNNs, such that the molecules can be interpreted as atoms with node features connected by edges carrying the bond information.

In many applications with molecules, however, we are more concerned with the overall properties of the molecules, such as solubility, polarity, etc., but it is not always straightforward to determine these quantities since some may require repeated and costly experiments, provided the molecules can be obtained in the first place. Alternatively, computational predictions based on the molecular structure and atom information are much cheaper and can be easily applied to a large batch of novel molecules. However, given the huge pool of possible molecules, it is infeasible to obtain a large set of labelled data for the model to generalise to all molecules for the downstream task.

In order to bypass the labelling issue, self-supervised learning(SSL) approaches have been employed to pre-train models for representation learning. It has been argued in [1] that this pretraining step can effectively equip the model with the capability of recognizing similar graph structures, resulting in better inferences. One simple SSL approach is constructing contrastive examples from existing

Preprint. Under review.

instances and have the model recognize the augmented pairs and maximize the agreement between them while differentiating each pair from others[2].

Following this augmentation-agreement-based approach, [1] proposed a reference for graph representation augmentation categories, including node dropping, edge perturbation, attribute masking, and sub-graphs. For molecules specifically, [3] explored atom masking, bond deletion and sub-graph removal to perturb a molecule in pretraining set for contrastive learning. Despite the promising results of better performance due to pretraining, there is one critical issue with this approach: perturbing molecular graphs, even very slightly, can significantly alter the physical and chemical properties of the underlying molecule, which defeats the purpose of representation pretraining for finetune tasks on the graph level.

Hence, we can make use of molecular scaffolds [4] as a basic unit such that all augmentations must preserve the underlying structures. Instead of randomly removing and masking atoms and bonds that can drastically change the molecule, we process the molecules with the same fundamental scaffold into pairs, and the goal of the pretraining therefore reduces to learning the representations of the scaffolds which are of higher significance when we are more concerned with the molecule-level predictions. Two main methods of such augmentations we peopose here are 1) clustering molecules of identical scaffolds and 2) constructing positive pairs based on a scaffold backbone.

Therefore, the main contributions of this paper are:

1. Implementing scaffold augmentation by clustering molecules of the identical scaffold structure
2. Generating synthetic molecules by adding atoms or smaller molecules to scaffold backbones
3. Comparing and analyzing the two methods for molecular scaffold augmentation with the established SSL pretraining procedure
4. Benchmarking the scaffold augmentation performance against previous augmentation baselines with downstream classification and regression tasks
5. Evaluating the GNN model performance with respect to simpler baseline models, as well as comparing different GNN models among themselves

2 Related Work

Graph Neural Networks (GNN) Graph Neural Networks (GNN) are powerful models for learning graph representations based on the graph structure and node information. Three essential types of GNNs - convolutional [5], attentional [6], and message-passing [7] - can be seen in almost all GNN layers, which characterise different mechanisms for aggregating information from neighboring nodes. A GNN layer can be interpreted as a permutation-equivariant function $\mathbf{F}(\mathbf{X}, \mathbf{A})$, and the general structure of a convolutional GNN can be formalised as

$$\mathbf{h_i} = \phi\big(\mathbf{x_i}, \oplus_{j \in \mathcal{N}_i} c_{i,j} \psi(\mathbf{x_j})\big),$$

where ϕ is the permutation-invariant propagation function, $\oplus$ some aggregation function, c_{ij} some normalization factor, and ψ a MLP function.

Graph (GIN) As an improvement of GCN layers, GIN layers have stronger expressive power [8], and can be written as

$$X^{k+1} = MLP_k\big(AX^k + (1 + \epsilon_k)X^k\big), \tag{1}$$

where A is the adjacency matrix, and ϵ_k a learnable parameter for each layer. Since we are interpreting each graph as a molecule and we require a graph-level representation for downstream tasks, a "readout" function is needed to aggregate node-level embeddings to produce a global embedding. This step for GIN can be shown as

$$h_G = CONCAT\big(READOUT_{v \in G}(\{h_v^{(k)}\}) | k = 0, 1..(K-1)\big). \tag{2}$$

One important feature is that node embeddings from previous layers are also accounted for so as to make better generalizations.

Molecular Scaffolds A molecular scaffold is a backbone structure that consists and characterises a group of molecules, and these molecules often exhibit similar synthetic pathways and chemical properties [4]. A number of examples can be found in [9].

Traditionally, Markush structures are used as scaffolds but they can be too generic for learning essential molecular features in applications. In the following implementations, however, Murcko scaffolds (BM) [10] are used to extract the bedrock structure, in which compounds are divided into a scaffold, linkers, R-groups, and ring systems [10], [11].

Self-supervised Learning(SSL) When data annotation becomes too costly for supervised learning, self-supervised learning(SSL) has the advantage of leveraging information from a very large set of unlabelled data, and it can be grouped into three main categories: generative, contrastive, and adversarial [2]. While generative methods usually employ a reconstruction loss and adversarial methods the distribution divergence, the contrastive architectures takes a similarity measure as the objective. Within contrastive learning, context-instance contrast and instance-instance contrast are the two main types, where the former attempts to capture key information of a local instance by associating with the global context, while the latter focuses on the instance-level representations, which is more relevant to many downstream tasks that do not require a "global understanding". In our `Scaffold` framework, we take the instance-instance view by having a pair-wise objective that only involve local similarity evaluations. This normalised temperature-scaled cross entropy loss (NT-Xent) [12] employed in the framework can be formulated as

$$\mathcal{L}_{i,j} = -\log \frac{\exp(sim((z_i, z_j))/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(sim(z_i, z_k))/\tau}$$

between examples i and j, where $sim(\mathbf{z}_i, \mathbf{z}_j)$ denotes the cosine similarity between the vectors.

3 Methods

The full `Scaffold` framework is split into three main components: augmentations, pretraining, and finetuning. We adapted our implementation of the pretraining and fine-tuning from the codebase [3], while the augmentations were implemented from scratch with two relevant approaches expanded below.

3.1 Scaffold Augmentation

3.1.1 Synthetic augmentation

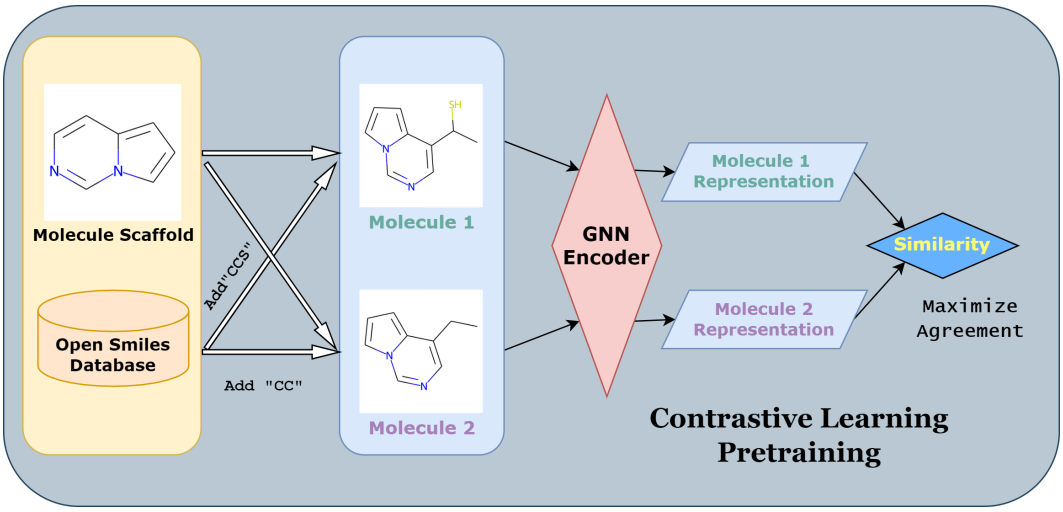


Figure 1: Synthetic augmentation contrastive learning pretraining framework

As seen in fig. 1, instead of removing or replacing existing atoms, we only perturb the scaffold by appending a smaller atom/molecule onto a random but plausible site on the scaffold. The overall procedure is listed here:

1. For each valid molecule, its BM scaffold is extracted and stored in the scaffold set S.
2. For each scaffold $s_i \in S$, we generate a certain number(N) pairs of augmented molecules, where each pair of molecule is produced by drawing two small molecules from the open SMILES database at random, and "appending" one molecule to each scaffold on a random site.
3. Store the generated molecule pairs as graphs (g_{i1}, g_{i2}).
4. During training, each molecule in the pair is encoded by a GNN encoder individually, and the consequent embeddings are used for the similarity metric(NT-Xent) to compute the loss $\mathcal{L}$.
5. The agreement between the two molecule embeddings is maximised by minimizing $\mathcal{L}$.

3.1.2 Clustering Augmentation

In synthetic augmentation, we need to acquire a large pool of existing general-purpose scaffolds and a set of small-molecules, and since the synthesis is conducted randomly disregarding the chemical feasibility, some generated molecules are very rare, if at all realistic, despite the theoretical correctness. This caveat may potentially bring extra noise to the augmentation data and render pretraining much less efficient. Although the all-purpose dataset may improve the robustness and generalisability of the pretrained model, it may not respond well to the downstream tasks when specific subsets of molecules are of concern. For example, for the BBBP task, pretraining only BBBP dataset could potentially prove more effective than pretraining on the full molecule dataset.

Therefore, we come up with a separate augmentation approach that only makes use of the existing molecules in the pretraining dataset. The procedure is outlined below:

1. For each molecule m_i, identify its scaffold s_j, and push it to the dictionary that maps known scaffolds S to a list of molecules with that scaffold.
2. Iterate over the known scaffolds and for each s_j, we randomly shuffle the list of molecules belonging to s_j into $\{m_1, m_2, ..., m_{2k}\}$, and then partition the list into two equal halves: $\{m_1, m_2, ..., m_k\}$ and $\{m_{k+1}, m_{k+2}, ..., m_{2k}\}$
3. Group the molecules by drawing m_i and m_{k+i} for each pair.

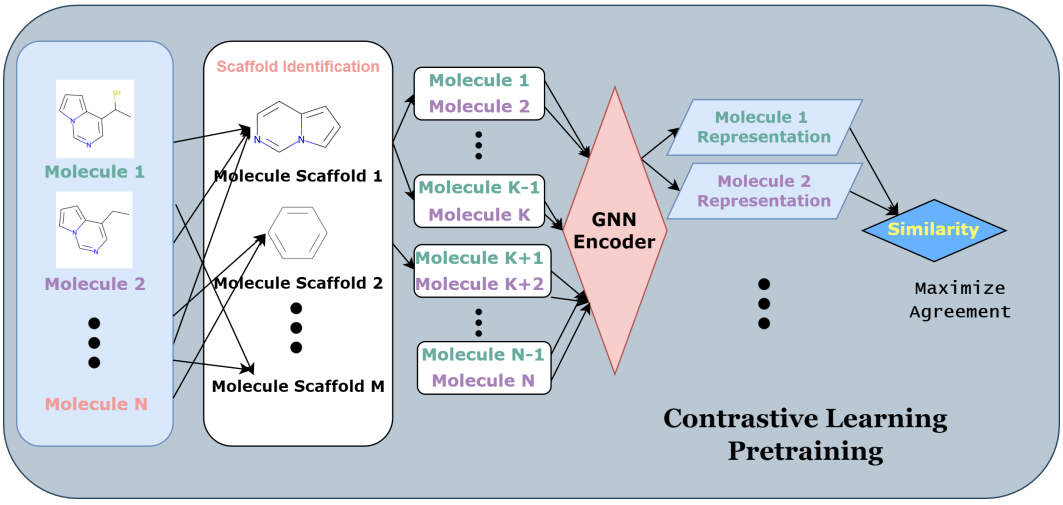


Figure 2: Clustering augmentation contrastive learning pretraining framework

With randomised pairs collected from all known scaffolds, the same procedure described in the previous section can be applied for contrastive pretraining.

3.2 GNN for graph-level inference

Our chief evaluation method is based on predictions of molecular properties, which include a wide variety of classification and regression tasks (section 5). Therefore, a finetuning procedure was set up for adjusting the pretrained models to the downstream tasks and a readout function was required (we used "MEAN" by default) to provide a single graph-level embedding which is then passed through an MLP head to predict the targets.

4 Datasets and Experiment Setup

In the synthetic augmentation procedure, One unlabelled scaffold dataset [3] was acquired from [13], together with a list of small molecules. Whereas for clustering augmentations, we take the raw data from datasets used by downstream tasks. Murcko scaffold identification and extraction in both approaches was conducted with the `Chem.Scaffolds.MurckoScaffold` module from rdkit. The same package was also used to "synthesize" extracted scaffolds with small molecules. The repository for the full `scaffold` augmentation implementation used by this paper can be found here.

A few representative datasets were selected so as to benchmark with the existing results and our own baseline methods. For classification tasks, we chose BBBP [14], BACE [15], Tox21 [16] and HIV [17]; for regression, FreeSolv [18] and ESOL [19] were used.

All finetuning tasks were performed with a single CPU, so that the training epoch for each task was limited to 30 due to the lack of resources. 100 epochs with 10 additional warm-up rounds were used in the one-time pretraining. The learning rate was set at 1e-4 for the GNN encoder and 5e-4 for the prediction head, and a weight-decay-regularized Adam optimizer was used throughout the experiments. Batch sizes were dynamically set based on the size of the dataset.

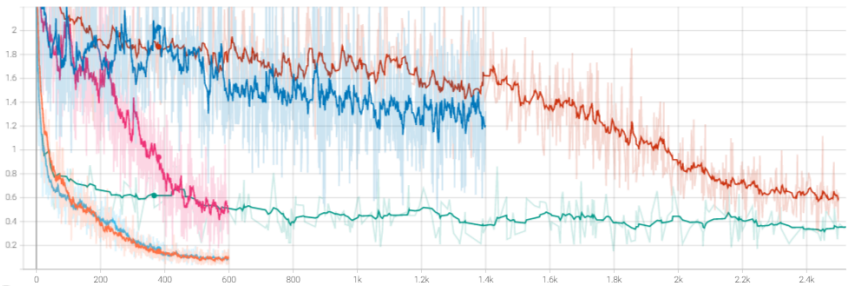
5 Evaluation and Discussion

Using the setup described above, we show the training curves for both clustering and synthesis based contrastive learning implementations in fig. 3. Fig. 3 presents 6 pretraining curves (each color represents a distinct dataset), and we can observe that in all cases, the training loss has steadily dropped, while the similar trend can be reflected in the validation loss as well. However, we noticed in some experiments, the GNN model had overfitted to the clustered pairs in training set, as the validation loss picked up with increasing epochs. Thus, we only keep the model with the best validation loss for finetuning purposes.

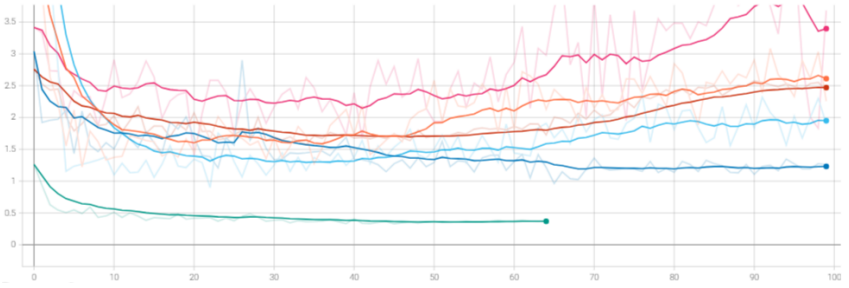
Similarly, fig. 4 displays the pretraining curve for synthetic augmentation pairs. Despite a spike in validation loss, towards the end of pretraining, there was still a downward trend, which indicated a higher capacity for the model to learn the scaffold latent representations.

We then evaluate the effect of pretraining by visualising the learned representations of one particular dataset. We take the clustering-augmentation example of ESOL data and first apply the pretrained model on m clustered pairs, resulting in $2m$ 256-dimensional embeddings. In order to make a scatter plot of the embeddings, we utilised PCA [20] for dimensionality reduction such that the 256-D vector can be projected onto a 2D plane, and the two orthogonal dimensions are chosen based on their "significance". In fig. 5a, we can see one principal cluster with a few breakaway molecules scattered away. Since each pair of molecule takes the same color, we can broadly observe that the representations within the same pair are mostly very close in the latent space. This finding can be more lucidly illustrated in fig. 5b, where each augmented pair is located far from each other, while molecules in the same pair are much closer in comparison. Hence, we can infer that the contrastive pretraining indeed enabled the GNN model to discern molecules of the same scaffold from the others.

With this premise demonstrated, we now show some finetuning results for the selected classification and regression tasks in fig. 6a and 6b. For each task, we benchmark the two augmentation method against the `random` baseline, which is a randomly initiated model without contrastive augmentations and pretraining. In the majority of the experiments, we can find a substantial improvement of the augmentation methods over the baseline with either higher ROC-AUC scores or lower RMSE. For instance, in `FreeSolv`, either augmentation approach provided an approximately 50% drop in RMSE.

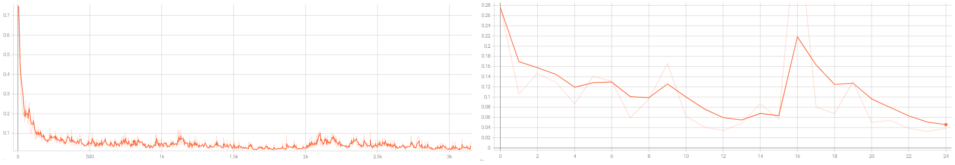


(a) Training loss



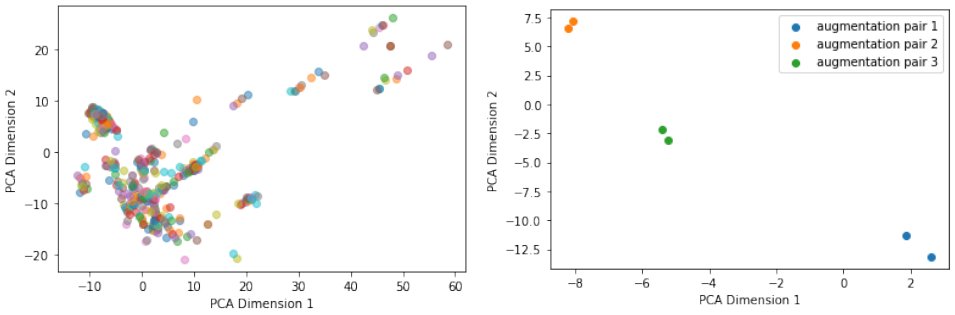
(b) Validation loss

Figure 3: Timeseries train and validation loss for Clustering Augmentation pretraining



(a) Training loss (b) Validation loss

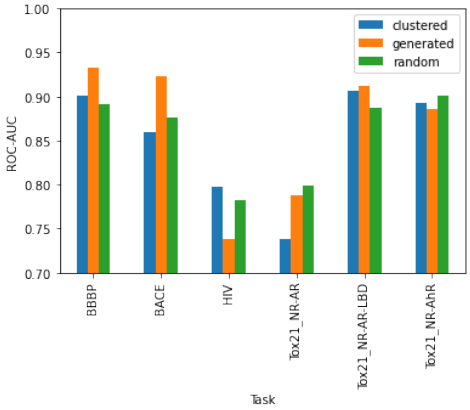
Figure 4: Timeseries train and validation loss for Synthetic Augmentation pretraining



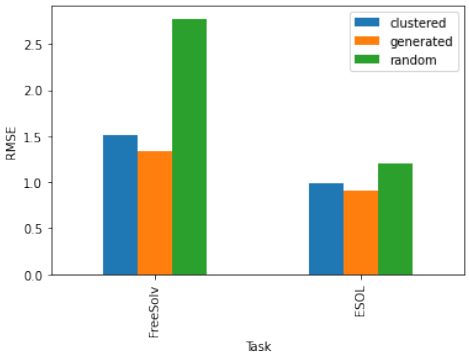
(a) All molecules (b) Randomly selected 3 pairs of molecules

Figure 5: Learned representations of ESOL molecules generated by pretrained clustering augmentation method projected from 256D to 2D plane with PCA. Each pair of augmented molecules are of the same color.

Nevertheless, in two of the three `Tox21` classification tasks, the `Random` baselines were marginally better than the rest. This can be largely attributed to the dataset itself due to the strong label imbalance (NR-AR with only 4.3% positive instances, and NR-AhR around 10.6%). Otherwise, synthetic method has a performance edge over clustered methods in general. One explanation for this disparity is that the synthetic augmentation approach was pretrained on a much larger dataset than its counterpart, and this data advantage gives the synthetically pretrained model a stronger



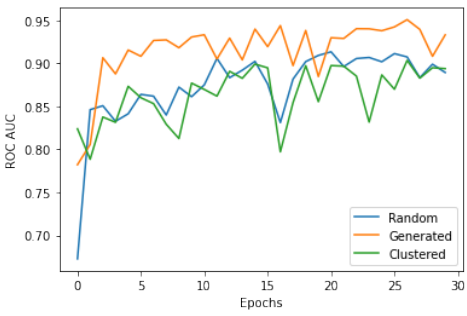
(a) Classification tasks ROC-AUC performance



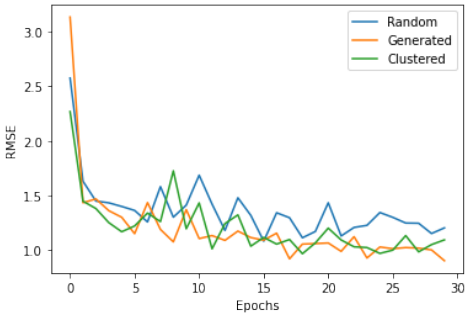
(b) Regression tasks RMSE performance

Figure 6: Finetune performances of both augmentation methods and the no-pretrain baseline for a selected set of classification and regression tasks.

capability for learning representations. This statement can be further corroborated in the `HIV` task, the only problem in which clustered method performed better. Unlike other datasets with merely a few thousand molecules, `HIV` consists of over 40k examples, thus empowering the clustering-pretrained model further.



(a) BBBP ROC-AUC metric over 30 epochs for all three methods



(b) ESOL RMSE metric over 30 epochs for all three methods

Figure 7: Evolution of validation performance during training for two selected tasks

Having established the efficacy of both augmentation methods, particularly the synthetic approach, we shift our focus from the test set ROC-AUC to another practical metric - training efficiency. This can be explored in two ways: (1) molecule augmentation efficiency based on running time and complexity and (2) finetuning validation result convergence rate. In terms of augmentation efficiency, the current implementation technically incurs a similar overhead for both methods, as each keeps a massive data structure to store the paired molecules for easier access during training. However, for the large pretraining dataset, the clustering-based method ran into out-of-memory error, and manifested a slower training speed in practice.

For evaluation metric convergence efficiency, we can plot time-series validation performance for both methods and the baseline to analyse the trends. The BBBP experiments were shown on fig. 7a, on which we can see that the synthetic augmentation approach outperforms the other two consistently, as expected. In addition, both augmentation methods had a significantly higher ROC-AUC than the baseline, verifying the validity of pretraining. Nonetheless, the baseline method quickly rose to the same level as the clustered method and continuously improved with the pretrained, thereby revealing an insignificant improvement of training efficiency. The almost identical findings can be repeated in the regression task from fig. 7b, but the synthetic augmentation curve suggested a steeper drop in RMSE, which prompts a higher training efficiency.

(a) BACE ROC-AUC metric over 30 epochs

(b) FreeSolv RMSE metric over 30 epochs

Figure 8: Comparison of GCN and GIN finetuning performance for two selected tasks over time

Further, we benchmark two distinct convolutional GNN models - GCN and GIN. In fig. 8a, we compare the validation ROC-AUC results for `BACE` classification task. Above all, in spite of the almost identical starting metric score, GIN indicated a much more efficient training process such that the ROC-AUC stabilised after only 7 epochs around the maximum level, while GCN saw a much slower increase over 30 epochs constantly below GIN. In fig. 8b, however, GIN model experienced much stronger variation early on, but its RMSE still dropped below that of GCN towards the end of 30 epochs. Overall, we notice that GCN improves slowly but more stably, yet GIN performs with stronger oscillations, higher efficiency, and eventually better end results. These conclusions also conform to the theoretical superiority of the expressive power of GIN over GCN, such that GIN can better distinguish geometrical structures that GCN might perceive as isomorphic.

Finally, we compare our results with existing GNN-based and traditional ML algorithms for some common tasks, shown in Table 1.

Model	ESOL	BBBP	BACE	Freesolv
RF	1.07	71.4	86.7	2.03
SVM	1.50	72.9	86.2	3.14
D-MPNN	0.98	71.2	85.3	2.18
MolCLR	1.11	73.8	89.0	2.20
SynAug	**0.90**	**93.3**	**92.3**	**1.33**
CluAug	0.98	90.1	86.0	1.51
Rand	1.20	89.1	87.6	2.78

Table 1: Test Performance of 7 models on 4 selected tasks. The first three methods are supervised (provided by [3]) while the last four are self-supervised pretrained models (with a rand baseline). ROC-AUC score is produced for classification tasks and RMSE for regression tasks.

Generally, pretrained models (MolCLR, SynAug, CluAug) manifest better results than GNNs without pretraining (Rand, D-MPNN), and the Rand GIN model performs better than message-passing and traditional ML methods. As expected, our SynAug performance trumps the others in all 4 selected tasks, which corroborates its efficacy in enabling the model to recognize molecular structures based on scaffolds. Meanwhile, CluAug comes very close to SynAug in all tasks as well, showing that pretraining on a much smaller but specialised dataset can also achieve significantly better results than No-pretraining baseline.

6 Future Work

In all experiments above, one recurring problem we have seen in many time-series plots is that the validation measures often had not appeared to reach an equilibrium and could still improve further.

This lack of training and early cutoff is due to the lack of training resources, as all experiments were conducted on one CPU. Furthermore, the clustering-augmentation pretraining could not be performed as a consequence of memory shortage.

Apart from the hardware challenges, some further experiments can be performed to provide more conclusive interpretations of the scaffold-based augmentations. Firstly, it would be helpful to quantify the faithfulness of the learned representations by, for instance, evaluating their mean Euclidean distance between pairs, and the measures for different approaches can be compared horizontally under similar conditions, which could provide a straightforward view on the effectiveness of the augmentation procedures. Secondly, a broader range of GNN-based algorithms can be evaluated with the current framework. Not only GNNs with higher expressive powers but also attentional and message-passing layers can be evaluated for the impact of the augmentation methods. In that case, we can show how and how much these augmentations can effectively improve the model capability in discerning molecules and learning their latent representations.

In terms of the augmentation implementations, a detailed study of how the method and agents of the "synthesis" can affect the pretraining and downstream tasks may help elucidate the principles of synthetic. Also, the implementation can be further optimised so that we can delegate the scaffold transformations to the training time to avoid memory usage issues.

References

[1] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations, 2020.

[2] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2021.

[3] Yuyang Wang, Jianren Wang, Zhonglin Cao, and Amir Barati Farimani. Molecular contrastive learning of representations via graph neural networks. *Nature Machine Intelligence*, 4(3):279–287, mar 2022.

[4] Candida Manelfi, Marica Gemei, Carmine Talarico, Carmen Cerchia, Anna Fava, Filippo Lunghini, and Andrea Rosario Beccari. "molecular anatomy": a new multi-dimensional hierarchical scaffold analysis tool. *Journal of Cheminformatics*, 13(1), jul 2021.

[5] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2016.

[6] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2017.

[7] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry, 2017.

[8] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks?, 2018.

[9] Hannes Stark. Self-supervised learning for small molecular graphs, 2021.

[10] Guy W. Bemis and Mark A. Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of Medicinal Chemistry*, 39(15):2887–2893, 1996. PMID: 8709122.

[11] Ye Hu, Dagmar Stumpfe, and Jürgen Bajorath. Computational exploration of molecular scaffolds in medicinal chemistry. *Journal of Medicinal Chemistry*, 59(9):4062–4076, 2016. PMID: 26840095.

[12] Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. On sampling strategies for neural network-based collaborative filtering, 2017.

[13] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. Pubchem 2019 update: improved access to chemical data. *Nucleic acids research*, 47(D1):D1102–D1109, 2019.

[14] Ines Filipa Martins, Ana L. Teixeira, Luis Pinheiro, and Andre O. Falcao. A bayesian approach to in silico blood-brain barrier penetration modeling. *Journal of Chemical Information and Modeling*, 52(6):1686–1697, 2012. PMID: 22612593.

[15] Govindan Subramanian, Bharath Ramsundar, Vijay Pande, and Rajiah Aldrin Denny. Computational modeling of b-secretase 1 (bace-1) inhibitors using ligand based approaches. *Journal of Chemical Information and Modeling*, 56(10):1936–1949, 2016. PMID: 27689393.

[16] Ruili Huang, Menghang Xia, Dac-Trung Nguyen, Tongan Zhao, Srilatha Sakamuru, Jinghua Zhao, Sampada A. Shahane, Anna Rossoshek, and Anton Simeonov. Tox21challenge to build predictive models of nuclear receptor and stress response pathways as mediated by exposure to environmental chemicals and drugs. *Frontiers in Environmental Science*, 3, 2016.

[17] Kaspar Riesen and Horst Bunke. Iam graph database repository for graph based pattern recognition and machine learning. In *SSPR/SPR*, 2008.

[18] David L. Mobley and J. Peter Guthrie. Freesolv: a database of experimental and calculated hydration free energies, with input files. *Journal of Computer-Aided Molecular Design*, 28(7):711–720, Jul 2014.

[19] John S. Delaney. Esol: Estimating aqueous solubility directly from molecular structure. *Journal of Chemical Information and Computer Sciences*, 44(3):1000–1005, 2004. PMID: 15154768.

[20] Andrzej Maćkiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342, 1993.