Learning the Neighborhood: Contrast-Free Self-Supervised Molecular Graph Pretraining

Boshra Ariguib[1] **Mathias Niepert**[1,2] **Andrei Manolache**[1,2,3]
ariguiba@studi.informatik.uni-stuttgart.de
mathias.niepert@ki.uni-stuttgart.de
andrei.manolache@ki.uni-stuttgart.de

[1]University of Stuttgart, Germany
[2]International Max Planck Research School for Intelligent Systems, Germany
[3]Bitdefender, Romania

Abstract

High-quality molecular representations are essential for property prediction and molecular design, yet large labeled datasets remain scarce. Self-supervised pretraining on molecular graphs has shown promise, but existing approaches often rely on costly negative sampling, hand-crafted augmentations, or complex generative and latent prediction objectives. We introduce C-FREE (***C**ontrast-**F**ree **R**epresentation learning on **E**go-ne**t**s*), a simple and effective framework that learns molecular representations by predicting subgraph embeddings from their complementary neighborhoods in the latent space. Motivated by the success of subgraph-based methods in supervised learning, C-FREE adopts fixed-radius ego-nets as the basic modeling unit and trains a hybrid Graph Neural Network (GNN)-Transformer backbone without negatives, positional encodings, or expensive pre-processing. Pretrained on the GEOM dataset, C-FREE achieves state-of-the-art performance on MoleculeNet, outperforming contrastive, generative, and more complex latent self-supervised learning techniques. Fine-tuning on the Kraken dataset further shows that pretraining on GEOM transfers effectively to new chemical domains, providing clear benefits over training from scratch. We make our code and best performing checkpoints publicly available at `https://github.com/ariguiba/C-FREE`.

1 Introduction and Related Work

High-quality molecular representations are critical for predicting properties, interpreting chemical behavior, and accelerating compound discovery [1, 2]. However, building such representations typically requires large labeled datasets, which are costly and scarce. Self-supervised learning (SSL) offers a promising alternative, and recent advances in vision and language modeling [3–9] have motivated its adaptation to molecular graphs. Broadly, existing approaches for graph self-supervised learning fall into three categories: *contrastive learning*, *generative pre-training*, and *latent representation learning*.

Contrastive learning aims to align representations of similar instances while pushing apart those of dissimilar ones, and it has been particularly influential in graph representation learning. For instance, GraphCL [10] adapts contrastive learning to graphs by designing diverse augmentations that capture structural priors, while JOAO [11] extends this idea by automatically selecting effective augmentations during training. Other approaches explore multi-view consistency, such as GraphMVP [12], which integrates 2D topology with 3D conformations, or InfoGraph [13], which maximizes mutual information between node- and graph-level embeddings. Although these methods produce transferable molecular representations [14], they are constrained by the need for negative samples and large

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Advancing Graph Machine Learning.

batch sizes [10]. In the case of graphs, the challenge is compounded by their irregular structures and varying sizes, which render naïve negative sampling unreliable and ambiguous, thereby the generation of non-trivial negatives often demands expensive computations.

Generative pre-training forms the second category of self-supervised learning, where models learn to reconstruct masked or missing components of a graph from the surrounding context. This objective encourages the capture of intrinsic structural and semantic properties. Early approaches include AttrMask [15], which predicts masked node attributes, and ContextPred [15], which trains GNNs to embed nodes occurring in similar structural contexts close together. EdgePred [16] extends this idea by predicting missing edges, while GPT-GNN [17] adopts an autoregressive formulation for full graph reconstruction. Building on these, GROVER [18] incorporates chemical domain knowledge by extracting molecular motifs and pre-training models to predict their presence. Despite their promise, generative approaches face the challenge of reconstructing both the discrete, sparse adjacency matrix and the node features, which may be continuous. Moreover, since graph nodes lack a natural ordering, it is often unclear how to define a valid starting point or sequence for autoregressive generation.

Finally, latent representation is the third category of self-supervised methods. Instead of reconstructing raw graph structures or features, these approaches predict target embeddings directly in the latent space. Operating in this space leverages compact, denoised, and semantically rich representations, often across different input modalities, and is generally easier than reconstructing the full graph, since only latent vectors need to be aligned rather than the entire adjacency matrix and feature set. Notable examples include BGRL [19], which adopts a bootstrapped strategy where an online encoder learns representations while a target encoder predicts outputs under different graph augmentations, and LaGraph [20] formulates self-supervised learning as latent graph prediction — because the latent graph itself is not available, it optimizes a computable upper bound of the prediction loss, combining reconstruction with invariance regularization applied only to masked nodes for more context-aware representations. While latent prediction methods avoid the costly generation of negative samples, their performance depends strongly on the quality of augmentations and the stability of model updates, as these methods are prone to representation collapse [21, 22]. Within latent representation learning, GraphJEPA [23] has recently extended the Joint Embedding Predictive Architecture (JEPA) [21] to graphs. GraphJEPA employs the computationally expensive METIS clustering algorithm to remove clusters and predict them from the remaining graph, generating patch-like substructures. It further encodes implicit hierarchical information by predicting subgraph coordinates on the unit hyperbola. While effective, this design introduces additional computational overhead and relies on auxiliary components—such as clustering, hierarchical encodings, and positional embeddings—that complicate training and may not be strictly necessary for learning useful representations.

Current work. To address these limitations, we introduce a self-supervised framework that adopts a non-contrastive predictive learning strategy with subgraphs as the basic modeling unit. Our approach is motivated by two goals: (i) *avoiding expensive or ambiguous design choices such as augmentations, subgraph extraction, and negative sampling*, where subgraph construction may require computationally heavy algorithms (e.g., METIS clustering [23]) and defining meaningful augmentations or negatives is non-trivial, since even molecules with nearly identical structures (e.g., chiral isomers) can exhibit very different properties, and (ii) *leveraging the success of subgraph-based methods in supervised learning* [24], which suggest that aggregating information from substructures can yield richer graph-level representations. Building on ideas from JEPA [21] and Equivariant Subgraph Aggregation Networks (ESAN) [24], our method generates subgraphs in a straightforward way and uses them with a non-contrastive predictive objective. We segment graphs into disjoint subgraphs, analogous to image patches or language tokens, and train the model to predict masked subgraphs from context. Unlike GraphJEPA and I-JEPA, our method avoids positional encodings, hierarchical objectives, and costly preprocessing such as clustering, relying instead on the inductive bias of GNNs and subgraph-based training to learn rich embeddings. Our contributions are as follows:

1. **A new pretraining task for molecular graphs.** We introduce a broadly applicable predictive objective based on k-EgoNet subgraphs, avoiding costly hand-crafted augmentations.

2. **A simple and effective training scheme.** We use non-contrastive predictive learning in place of contrastive objectives, eliminating the pre-train/fine-tune mismatch and removing the need for negative pairs or heavy augmentations. Moreover, we show that our fine-tuning can simulate ESAN [24] and is provably strictly more expressive than 1-WL [25].

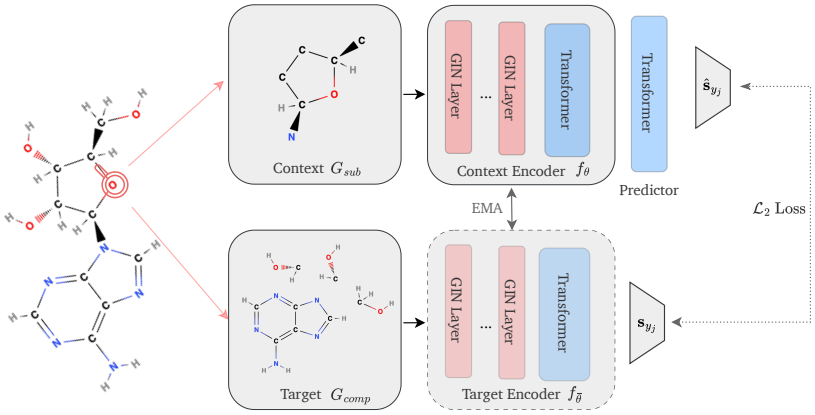

Figure 1: From each molecular graph, we construct complementary context and target subgraphs by sampling a random node and extracting its k-EgoNet [24] with $k \in \{2, 3, 4\}$ to obtain sufficiently large neighborhoods. Both subgraphs are independently encoded; the context embedding is passed through a predictor to estimate the target embedding. We pool the predicted and encoded target embeddings and minimize their mean squared $\mathcal{L}_2$ loss. To prevent representation collapse [22, 21], the target encoder is maintained as an exponential moving average (EMA) of the context encoder.

3. **State-of-the-art results.** Our approach matches or surpasses other self-supervised models, achieving the best average performance on MoleculeNet [26], and showing strong transfer capabilities to novel molecular datasets such as Kraken [27].

2 Contrast-Free Self-Supervised Pretraining

In the following, we outline our proposed training pipeline, illustrated in Fig. 1. Unlike most generative methods [15, 16], we apply our training objective fully in the latent space, without reconstructing the original features of the masked components. The core principle of our approach is to learn representations by predicting the embedding of one view of the data, denoted as the target, from the embedding of another, related view, denoted as the context.

Specifically, we represent a molecule as a graph $G = (V, E)$ where V is the set of nodes (e.g., atoms) and E is the set of edges (e.g., covalent bonds). From G, we sample a subgraph $G_{sub} = (V_{sub}, E_{sub})$ and its complementary graph $G_{comp} = (V \setminus V_{sub}, E \setminus E_{sub})$. To construct G_{sub}, we sample a random node v_i and generate its k-EgoNet $E(v_i)$, defined as the k-hop neighborhood including all induced edges [24]; the remaining nodes and edges form G_{comp}. The model then predicts the embedding of the target subgraph from that of its associated context subgraph. This design is loosely inspired by ESAN [24], but adopts a simplified variant: we use fixed-radius ego-nets as complementary views during pretraining, and at fine-tuning we evaluate both linear probing on whole-graph embeddings and an aggregation of subgraph embeddings using DeepSets [28].

Context–Target View Generation. We generate complementary views by sampling k-EgoNets, where the k-hop neighborhood of a node defines one subgraph and the remaining nodes and edges define its complement (see Fig. 2). Either view can serve as the target while the other acts as context, and their roles are alternated during training to avoid prediction bias. We adopt fixed-radius neighborhoods with $k \in \{2, 3, 4\}$, analogous to fixed-size patches in vision-based methods [21]. Although graphs vary in size and structure, this ensures that each subgraph captures a comparable amount of local information. To further diversify training, we sample multiple nodes $v_1, v_2, \ldots, v_n$ per molecule and construct their corresponding k-EgoNets $E(v_1), E(v_2), \ldots, E(v_n)$, yielding multiple complementary context–target pairs without increasing dataset size.

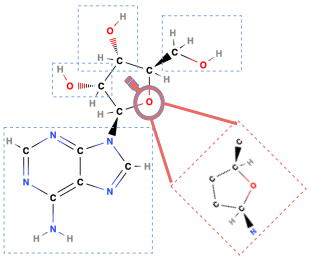

Figure 2: To generate subgraphs, we sample a random node from the original graph (here, the oxygen atom) and extract its 2-EgoNet as the context subgraph (outlined by red square). The remaining components (outlined by blue squares) constitute the target subgraph.

Context Encoder. We aim to learn subgraph representations that generalize effectively to whole-molecule embeddings. Following the architecture proposed in [29], we use a message-passing neural network (MPNN) with GINE [15, 30] as the backbone to capture local structural information and stack a Transformer module with multiple self-attention layers on top to capture global dependencies. For the final embedding, for each node, we pool its intermediate representations obtained from the GINE layers, and then pass the node embeddings to the Transformer module as tokens.

Predictor Network. The predictor takes the context subgraph representation and learns to predict the embedding of its complementary subgraph. It is implemented as a lightweight transformer, with stacked attention layers followed by an MLP. In our architecture, structural information is already captured implicitly by the MPNN-based context encoder, so unlike image-based JEPA [21] and GraphJEPA [31], we do not rely on explicit positional encodings.

Target Encoder. The target subgraph $f_{\bar{\theta}}$ is encoded by a separate instance of the context encoder. Maintaining two distinct networks stabilizes training and mitigates representation collapse, a strategy widely adopted in self-predictive frameworks such as BYOL [22], I-JEPA [21], and BGRL [19]. The target encoder's weights are updated via an exponential moving average (EMA) of the context encoder's parameters:

$$\bar{\theta}^{(t)} = \tau\, \bar{\theta}^{(t-1)} + (1-\tau)\, \theta^{(t)}$$

where $\bar{\theta}^{(t)}$ are the exponentially moving averaged parameters at step t, $\theta^{(t)}$ are the current parameters, and $\tau \in [0, 1]$ is the decay rate controlling the contribution of past parameters.

Pretraining task. We obtain a single embedding for each subgraph by pooling the final node embeddings. For the context subgraph we use the outputs of the predictor, and for the target subgraph we use the outputs of the encoder. The self-supervised pretraining objective is to minimize the mean squared $\mathcal{L}_2$ distance between the predicted and target subgraph embeddings:

$$\frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{k} \left\| \hat{\mathbf{s}}_{y_j} - \mathbf{s}_{y_j} \right\|^2$$

where $\hat{\mathbf{s}}y_j$ and $\mathbf{s}y_j$ denote the predicted and target subgraph embeddings, M is the batch size, and k the number of sampled views (ego-nets and their complements). All views are treated as separate instances when computing the loss.

Fine-tuning. When fine-tuning for downstream tasks, we use the target encoder as our pretrained backbone to generate graph embeddings, and add lightweight task-specific heads. We consider two types of task heads: (i) linear probing on whole-graph embeddings by using a single linear layer (C-FREE$_{\text{LIN}}$) to evaluate the quality of the representations on downstream tasks and (ii) aggregating the k-EgoNet subgraph embeddings with DeepSets [28] (C-FREE$_{\text{DS}}$), demonstrating that subgraph pretraining transfers both to whole-molecule prediction and to ESAN-style fine-tuning schemes.

2.1 Expressiveness

Finally, we make a simple theoretical observation: when using the DeepSets head, C-FREE$_{\text{DS}}$ simulates ESAN [24] and is strictly more expressive than the 1-WL algorithm [25]. The formal text and proof for the Lemma is deferred to Appendix Section 5.1.

(Informal) Lemma 1. *Under the assumptions from Theorem 2 of [24], C-FREE with a DeepSets task head is as expressive as ESAN, hence it is strictly more expressive than the* 1*-WL algorithm [25].*

3 Empirical Evaluation

We evaluate our framework through four complementary sets of experiments:

(i) We compare against both contrastive and non-contrastive self-supervised methods on the MoleculeNet [26] benchmark, which consist of classification tasks, using a frozen backbone to assess the quality of the learned representations (Section 3.1).

(ii) We examine whether pretraining accelerates convergence and improves downstream performance compared to random initialization by fully fine-tuning on Kraken [32], a dataset focused on molecular property regression (Section 3.2).

Table 1: Performance comparison on molecular property prediction tasks from the MoleculeNet [26] datasets. The feature extractor backbone is frozen. **Non-CL** refers to non-contrastive methods, and **CL** refers to contrastive methods. Our model is reported as **C-FREE$_{\text{LIN}}$**, which uses whole-molecule embeddings with a linear probe for fine-tuning, and **C-FREE$_{\text{DS}}$**, which aggregates k-EgoNet subgraph embeddings with DeepSets [28]. The evaluation metric is ROC-AUC (↑). **Red** highlights the best model and **Blue** the second best. **C-FREE** achieves the best or second-best results on 4 out of 7 datasets, with **C-FREE$_{\text{DS}}$** ranking first overall and **C-FREE$_{\text{LIN}}$** being the second best model.

		MoleculeNet Datasets							
		BBBP (↑)	Tox21 (↑)	ToxCast (↑)	Sider (↑)	ClinTox (↑)	HIV (↑)	Bace (↑)	Avg (↑)
	Random init.	$50.7_{\pm 2.5}$	$64.9_{\pm 0.5}$	$53.2_{\pm 0.3}$	$53.2_{\pm 1.1}$	$63.1_{\pm 2.3}$	$66.1_{\pm 0.7}$	$63.4_{\pm 1.8}$	59.2
CL	InfoGraph	$65.9_{\pm 0.6}$	$65.8_{\pm 0.7}$	$54.6_{\pm 0.1}$	$57.2_{\pm 1.0}$	$61.4_{\pm 4.8}$	$71.4_{\pm 0.6}$	$67.4_{\pm 4.9}$	63.4
CL	GROVER	**67.0**$_{\pm 0.3}$	$63.9_{\pm 0.3}$	$53.6_{\pm 0.4}$	**59.9**$_{\pm 1.7}$	$65.0_{\pm 6.4}$	$67.8_{\pm 1.0}$	$69.0_{\pm 4.7}$	63.7
CL	GraphCL	$64.7_{\pm 1.7}$	$69.1_{\pm 0.5}$	$56.2_{\pm 0.2}$	**59.5**$_{\pm 0.9}$	$60.8_{\pm 3.0}$	$72.5_{\pm 1.4}$	**77.0**$_{\pm 1.7}$	65.7
CL	JOAO	$66.1_{\pm 0.8}$	$68.1_{\pm 0.2}$	$55.1_{\pm 0.4}$	$58.3_{\pm 0.3}$	$65.3_{\pm 6.1}$	**73.8**$_{\pm 1.2}$	$71.1_{\pm 0.8}$	65.4
CL	GraphMVP	**69.2**$_{\pm 1.8}$	$63.8_{\pm 0.3}$	$55.5_{\pm 0.3}$	$58.6_{\pm 0.4}$	$58.7_{\pm 1.9}$	$68.6_{\pm 1.0}$	$73.3_{\pm 4.7}$	64.0
Non-CL	EdgePred	$54.2_{\pm 1.0}$	$66.2_{\pm 0.2}$	$54.4_{\pm 0.1}$	$56.1_{\pm 0.1}$	$65.4_{\pm 5.0}$	**73.6**$_{\pm 0.4}$	$71.4_{\pm 1.2}$	63.0
Non-CL	AttrMask	$62.7_{\pm 2.7}$	$65.7_{\pm 0.8}$	$56.1_{\pm 0.2}$	$58.3_{\pm 1.5}$	$61.9_{\pm 6.4}$	$65.5_{\pm 1.4}$	$64.8_{\pm 2.6}$	62.1
Non-CL	GPT-GNN	$62.0_{\pm 0.9}$	$64.9_{\pm 0.7}$	$55.4_{\pm 0.2}$	$55.3_{\pm 0.8}$	$55.0_{\pm 5.1}$	$71.2_{\pm 1.5}$	$61.0_{\pm 1.2}$	60.7
Non-CL	Cont. Pred	$55.5_{\pm 2.0}$	$67.9_{\pm 0.7}$	$54.0_{\pm 0.3}$	$57.1_{\pm 0.5}$	**67.4**$_{\pm 4.3}$	$66.2_{\pm 1.5}$	$54.4_{\pm 3.2}$	60.4
	C-FREE$_{\text{LIN}}$	$60.5_{\pm 1.7}$	**76.1**$_{\pm 0.2}$	**62.7**$_{\pm 0.4}$	$59.0_{\pm 0.6}$	$62.7_{\pm 1.0}$	$68.7_{\pm 0.4}$	**75.8**$_{\pm 0.9}$	**66.6**
	C-FREE$_{\text{DS}}$	$64.2_{\pm 3.8}$	**76.7**$_{\pm 0.6}$	**63.9**$_{\pm 0.3}$	$58.0_{\pm 0.7}$	**71.4**$_{\pm 3.7}$	$65.5_{\pm 0.6}$	$73.9_{\pm 0.7}$	**67.7**

(iii) We conduct an ablation study on the predictor network to determine its impact on representation quality and training stability (Section 3.3).

(iv) We verify whether the empirical expressiveness aligns with the theoretical result from (Informal) Lemma 1 (Appendix Section 5.1).

Implementation details for pretraining and evaluation are provided in Section 5.5 in the Appendix.

3.1 Comparison with Frozen Backbones

For the first set of experiments, we compare our framework against state-of-the-art contrastive and non-contrastive self-supervised methods in the transfer learning setting on molecular property classification tasks. Following the protocol of Wang et al. [14], we pre-train a backbone on all eligible molecules from the GEOM dataset [33] (about 0.33M) and use the resulting embeddings on MoleculeNet [26] classification tasks. We report results for two evaluation strategies: applying a linear probe directly on whole-graph embeddings to test generalization, and applying a linear probe on aggregated subgraph embeddings combined with DeepSets [24]. Performance is reported as the mean and standard deviation of ROC-AUC scores over three scaffold splits, with test scores taken from the model achieving the best validation performance.

As shown in Table 1, our framework achieves the best average performance across all datasets and outperforms baselines on 4 of the 7 tasks. The gain from using DeepSets-aggregated subgraphs is consistent with our pretext task design, though this comparison should be interpreted with caution since the DeepSets variant introduces additional parameters into the linear probe. Notably, our method performs particularly well when compared to other non-contrastive approaches, highlighting the effectiveness of our predictive learning strategy. We also observe strong gains on multi-task classification datasets such as Tox21 (12 tasks), ToxCast (617 tasks), and Sider (27 tasks), suggesting that our method may capture more generalizable features across related prediction tasks.

3.2 Full Fine-tuning for Property Regression

Having established a comparison with a frozen backbone on classification tasks, we next examine the transfer capabilities of our pretrained model in a regression setting. For this, we fine-tune the backbone pretrained on GEOM [33] end-to-end on the Kraken dataset [27] and add a separate 2-layer MLP head for each target, without using DeepSets. Kraken contains 1,552 ligands labeled with four 3D descriptors: Sterimol B5, Sterimol L, buried Sterimol B5, and buried Sterimol L. These molecules are not seen during pretraining and provide a strong test of generalization. Although Kraken includes DFT-computed conformer ensembles, we do not train on conformers and only use 2D graphs.

As shown in Fig. 3, two trends are clear: (i) models initialized with GEOM pretraining start with substantially lower MAE than randomly initialized counterparts, and (ii) even after 250 epochs,

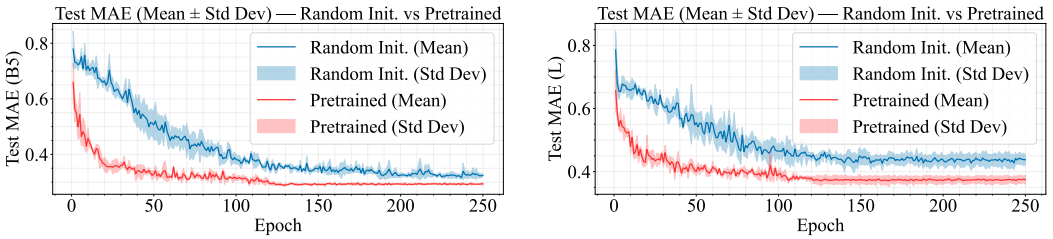

Figure 3: Test MAE on Kraken regression tasks (Sterimol B5 and Sterimol L) comparing random initialization and GEOM-pretrained models. Pretrained models start with lower error and converge faster, while randomly initialized models fail to match their performance even after 250 epochs. Curves show the mean over 3 runs, with shaded regions indicating the standard deviation.

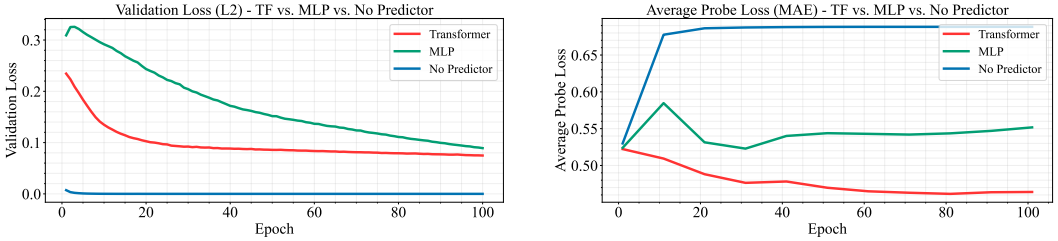

Figure 4: Predictor ablation. Left: SSL validation loss on GEOM for the pretraining predictive task. Right: average linear-probe (MAE ↓) on Kraken using frozen backbones. Training without a predictor collapses (loss∼0) and yields the worst probes; an MLP predictor helps but underperforms; a Transformer predictor delivers the best downstream performance.

random initialization fails to match the performance of pretrained models. Results for BurB5 and BurL are included in the Appendix Fig. 6.

3.3 Ablation on Predictor Types

We hypothesize that our model's strong performance stems from the predictor network, which serves as a guiding signal to refine the representations produced by the encoder. To test this, we pretrain on GEOM and perform an ablation with three predictor variants: none, a linear predictor, and a transformer. We then evaluate downstream performance using either a linear probe or full fine-tuning.

As shown in Appendix Table 8, removing the predictor leads to poor downstream performance across all regression tasks, consistent with representation collapse, as the self-supervised loss quickly converges to zero (see Fig. 4). While the target encoder is updated via Exponential Moving Average to stabilize training [34, 22, 21], this alone is insufficient: without an asymmetric architecture, the model collapses to a trivial solution. Introducing a predictor breaks this symmetry and is hypothesized to empirically prevent collapse [35]. Even a simple two-layer MLP improves performance, while a transformer-based predictor yields the strongest results. Unlike the MLP, where we pool node embeddings before prediction, the transformer predicts at the node level prior to pooling, potentially producing more informative graph-level representations; we therefore adopt it as the default.

4 Conclusions and Future Work

We presented C-FREE, a simple yet effective framework for molecular representation learning that predicts subgraph embeddings from their complementary neighborhoods in the latent space. Our method consistently outperforms both contrastive and non-contrastive baselines while avoiding costly overhead such as positional encodings or complex subgraph partitioning algorithms. We further showed that incorporating the transformer architecture in both the encoder and predictor strengthens representation quality and improves training stability. Importantly, C-FREE demonstrates strong transfer to unseen molecules and new tasks, including classification benchmarks such as MoleculeNet and regression datasets such as Kraken, underlining its promise as a strong foundation model for molecular learning.

There are several natural extensions of our work. First, improving the subgraph selection process to incorporate more chemically meaningful or substructure-aware strategies may lead to richer

representations. Second, since our framework builds on the modular MolMix backbone [29], it can be readily extended to multimodal inputs by adding encoders for different data types, enabling joint training on 2D molecular graphs together with SMILES strings and 3D conformers. Finally, exploring alternative pretraining objectives and transfer strategies, such as different ways of aggregating information across subgraphs, could further improve performance. Beyond chemistry, applying our pretraining strategy to other graph domains, including citation networks or knowledge graphs, would provide a broader test of its applicability.

References

[1] Daniel S. Wigh, Jonathan M. Goodman, and Alexei A. Lapkin. A review of molecular representation in the age of machine learning. *WIREs Comput. Mol. Sci.*, 12(5):e1603, September 2022. ISSN 1759-0876. doi: 10.1002/wcms.1603.

[2] Daniel C. Elton, Zois Boukouvalas, Mark D. Fuge, and Peter W. Chung. Deep learning for molecular design—a review of the state of the art. *Mol. Syst. Des. Eng.*, 4(4):828–849, August 2019. ISSN 2058-9689. doi: 10.1039/C9ME00039A.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Guide Proceedings*, volume 119, pages 1597–1607. JMLR.org, July 2020. doi: 10.5555/3524938.3525087.

[5] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf`.

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jegou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9630–9640, 2021. doi: 10.1109/ICCV48922.2021.00951.

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, July 2021.

[8] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[9] Yann LeCun and Courant. A path towards autonomous machine intelligence. 2022. URL `https://api.semanticscholar.org/CorpusID:251881108`.

[10] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph Contrastive Learning with Augmentations, April 2021.

[11] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph contrastive learning automated. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12121–12132. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/you21a.html`.

[12] Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. Pre-training Molecular Graph Representation with 3D Geometry, May 2022.

[13] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization, January 2020.

[14] Hanchen Wang, Jean Kaddour, Shengchao Liu, Jian Tang, Joan Lasenby, and Qi Liu. Evaluating self-supervised learning for molecular graph embeddings. *Advances in Neural Information Processing Systems*, 36:68028–68060, 2023.

[15] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for Pre-training Graph Neural Networks, February 2020.

[16] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[17] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. Gpt-gnn: Generative pre-training of graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1857–1867, 2020.

[18] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying WEI, Wenbing Huang, and Junzhou Huang. Self-Supervised Graph Transformer on Large-Scale Molecular Data. In *Advances in Neural Information Processing Systems*, volume 33, pages 12559–12571. Curran Associates, Inc., 2020.

[19] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L. Dyer, Rémi Munos, Petar Veličković, and Michal Valko. Large-Scale Representation Learning on Graphs via Bootstrapping, February 2023.

[20] Yaochen Xie, Zhao Xu, and Shuiwang Ji. Self-Supervised Representation Learning via Latent Graph Prediction. In *Proceedings of the 39th International Conference on Machine Learning*, pages 24460–24477. PMLR, June 2022.

[21] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-Supervised Learning From Images With a Joint-Embedding Predictive Architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15619–15629, 2023.

[22] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020.

[23] Geri Skenderi, Hang Li, Jiliang Tang, and Marco Cristani. Graph-level Representation Learning with Joint-Embedding Predictive Architectures, January 2025.

[24] Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan, Michael M. Bronstein, and Haggai Maron. Equivariant Subgraph Aggregation Networks, March 2022.

[25] Boris Weisfeiler and Andrei Leman. The reduction of a graph to canonical form and the algebra which appears therein. *nti, Series*, 2(9):12–16, 1968.

[26] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[27] Yanqiao Zhu, Jeehyun Hwang, Keir Adams, Zhen Liu, Bozhao Nan, Brock Stenfors, Yuanqi Du, Jatin Chauhan, Olaf Wiest, Olexandr Isayev, Connor W. Coley, Yizhou Sun, and Wei Wang. Learning Over Molecular Conformer Ensembles: Datasets and Benchmarks, July 2024.

[28] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

[29] Andrei Manolache, Dragos Tantaru, and Mathias Niepert. MolMix: A Simple Yet Effective Baseline for Multimodal Molecular Representation Learning, October 2024.

[30] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks?, February 2019.

[31] Geri Skenderi, Hang Li, Jiliang Tang, and Marco Cristani. Graph-level Representation Learning with Joint-Embedding Predictive Architectures. October 2023.

[32] Tobias Gensch, Gabriel dos Passos Gomes, Pascal Friederich, Ellyn Peters, Théophile Gaudin, Robert Pollice, Kjell Jorner, AkshatKumar Nigam, Michael Lindner-D'Addario, Matthew S Sigman, et al. A comprehensive discovery platform for organophosphorus ligands for catalysis. *Journal of the American Chemical Society*, 144(3):1205–1217, 2022.

[33] Simon Axelrod and Rafael Gomez-Bombarelli. Geom, energy-annotated molecular conformations for property prediction and molecular generation. *Scientific Data*, 9(1):185, 2022.

[34] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/68053af2923e00204c3ca7c6a3150cf7-Paper.pdf`.

[35] Pierre H. Richemond, Jean-Bastien Grill, Florent Altché, Corentin Tallec, Florian Strub, Andrew Brock, Samuel Smith, Soham De, Razvan Pascanu, Bilal Piot, and Michal Valko. Byol works even without batch statistics, 2020. URL `https://arxiv.org/abs/2010.10241`.

[36] Ralph Abboud, İsmail İlkan Ceylan, Martin Grohe, and Thomas Lukasiewicz. The surprising power of graph neural networks with random node initialization, 2021. URL `https://arxiv.org/abs/2010.01179`.

[37] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1):359–392, 1998.

[38] Kristof T. Schütt, Pieter-Jan Kindermans, Huziel E. Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions, December 2017.

[39] Shikun Feng, Yuyan Ni, Minghao Li, Yanwen Huang, Zhi-Ming Ma, Wei-Ying Ma, and Yanyan Lan. UniCorn: A Unified Contrastive Learning Approach for Multi-view Molecular Representation Learning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 13256–13277. PMLR, July 2024.

[40] Taojie Kuang, Yiming Ren, and Zhixiang Ren. 3D-Mol: A Novel Contrastive Learning Framework for Molecular Property Prediction with 3D Information, June 2024.

[41] Kha-Dinh Luong and Ambuj K. Singh. Fragment-based Pretraining and Finetuning on Molecular Graphs. *Advances in Neural Information Processing Systems*, 36:17584–17601, December 2023.

[42] Hengrui Zhang, Qitian Wu, Junchi Yan, David Wipf, and Philip S Yu. From Canonical Correlation Analysis to Self-supervised Graph Neural Networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 76–89. Curran Associates, Inc., 2021.

[43] Namkyeong Lee, Junseok Lee, and Chanyoung Park. Augmentation-Free Self-Supervised Learning on Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7): 7372–7380, June 2022. ISSN 2374-3468. doi: 10.1609/aaai.v36i7.20700.

Table 2: Expressiveness experiment results on the EXP [36] dataset. Our C-FREE framework outperforms standard GNNs, with the 2- and 3-EgoNet variants achieving the highest accuracy. In particular, they surpass GraphJEPA, which relies on the computationally expensive METIS [37] clustering algorithm to extract substructures, as well as a standard MPNN baseline (GINE).

METHOD	ACCURACY (↑)
GINE [30]	$50.69_{\pm 1.39}$
GRAPHJEPA [23]	$98.77_{\pm 0.99}$
C-FREE (1-EGONET)	$96.03_{\pm 1.22}$
C-FREE (2-EGONET)	$99.33_{\pm 0.18}$
C-FREE (3-EGONET)	$99.08_{\pm 0.20}$

5 Supplementary Materials

5.1 Expressiveness

Here, we provide the proof for (Informal) Lemma 1 in the main paper.

Lemma 1. *Let C-FREE$_{DS}$ be a model as defined in Section 2 with a subgraph encoder f_θ consisting of a 1-WL MPNN (e.g., GIN/GINE) followed by a Transformer without positional encodings, and a DeepSets task head DS. For any k-EgoNet policy with $k \geq 1$ under the assumptions of Theorem 2 from [24], C-FREE$_{DS}$ is as expressive as ESAN [24] with an EGO policy, therefore it is as most as expressive as DS-WL and strictly more expressive than 1-WL.*

Proof. Fix a k-EGO policy $\pi = \text{EGO}_k$ with $k \geq 1$ and let $S_\pi(G)$ be the multiset of k-ego-nets with their complements (edge-covering in the ESAN sense). Define:

$$f_{C\text{-}FREE}(G) = DS\big(\{f_\theta(S) : S \in S_\pi(G)\}\big),$$

Due to Theorem 1 of [29], we have that f_θ maintains the permutation equivariance of the MPNN; moreover, since there exists a parametrization of the Transformer that can approximate the identity map arbitrarily well, the Transformer does not lower the expressive power of the MPNN. We therefore have that f_θ is as powerful as 1-WL.

Since we have a DeepSets encoder DS and an edge-covering k-EGO policy, we can use the same proof argument as in Theorem 2 from [24], i.e. we apply f_θ to each $S \in S_\pi(G)$ and then aggregate the multisets with DS, therefore $f_{C\text{-}FREE}$ simulates ESAN, and is at most as expressive as *DS-WL* and strictly more expressive than 1-WL.

□

To validate our theoretical findings, we run an experiment to examine the expressive power of our framework. We use the EXP dataset, which is specifically designed by Abboud et al. [36] so that any 1-WL GNN cannot do better than random guess. We design an end-to-end training experiment with a smaller version of our backbone and compare the results to a Vanilla GINE [15] network and to GraphJEPA [21]. We employ a 3-layer GNN encoder followed by a 2-layer transformer with 2 attention heads, using a hidden dimension of 96 throughout. Results are averaged over three runs with resampled EgoNets. As shown in Table 2, even the 1-EgoNet variant of our C-FREE framework achieves strong performance, approaching the theoretical upper bound. Both the 2- and 3-EgoNet variants further improve accuracy, outperforming standard GNNs such as GINE and GraphJEPA, the latter relying on the computationally expensive METIS [37] algorithm.

5.2 Computational Complexity Analysis

For generating the subgraphs used as input units in our pre-training scheme, we employ k-EgoNets with fixed radii $k \in \{3, 4\}$. We extract k-hop neighborhoods using PyTorch Geometric's [**?**] `k_hop_subgraph` function, which performs a breadth-first search (BFS) from each node and collects all nodes reachable within k hops. For constant k, the BFS cost is bounded by the number of explored edges, yielding a worst-case complexity of $O(|E|)$. When repeated for all k radii, this results in $O(k \cdot |E|)$, where $|E|$ denotes the total number of edges. In practice, the number of explored edges is proportional to the average degree d, giving a total cost of $O(k \cdot d \cdot |V|)$, where $|V|$ is the number

Table 3: Average runtime (in milliseconds) for generating a single subgraph on the GEOM dataset, comparing METIS partitions with $n \in \{16, 32\}$ patches and k-EgoNets with $k \in \{3, 4\}$.

	METHOD	AVG. RUNTIME (MS)
METIS	N = 32	1.123
	N = 16	1.031
K-EGONETS	K = 3	0.171
	K = 4	0.185

Table 4: Computational efficiency of different SSL methods from [14], showing the number of trainable parameters for each backbone. We report both the total parameters of our backbone and those of the encoder alone, since only the latter is used for downstream evaluation. By discarding nearly half of the backbone parameters in this stage, our approach remains competitive without increasing the parameter count for downstream tasks, further highlighting its efficiency.

METHOD	#PARAMETERS (MILLION)
EDGEPRED	7.46
ATTRMASK	7.61
GPT-GNN	7.61
INFOGRAPH	7.82
GROVER	7.57
CONT.PRED	12.00
GRAPHCL	8.19
JOAO	8.19
GRAPHMVP	15.84
C-FREE (FULL)	8.09
C-FREE (ENCODER)	4.67

of nodes. Since molecular graphs are sparse, neighborhood growth is modest: an analysis of the GEOM dataset used for pre-training shows that the average degree is only $d = 2.1$. As a result, k-hop neighborhoods remain small, and `k_hop_subgraph` is computationally efficient, scaling linearly with the number of nodes $O(|V|)$ in practice.

In comparison, the METIS algorithm [37] used in GraphJEPA [23] employs a multilevel graph partitioning approach. While the algorithm is often cited with an overall complexity of $O(|E| \cdot \log |V|)$ in practice, it consists of three main phases: (1) a coarsening phase that uses heavy-edge matching to successively reduce the graph size, (2) an initial partitioning phase that partitions the smallest coarsened graph (with negligible complexity due to its small size), and (3) an uncoarsening/refinement phase that projects the partition back to the original graph while refining it at each level. The coarsening and refinement phases dominate the computational cost, each contributing $O(|E| \cdot \log |V|)$ complexity. However, since molecular graphs are sparse, similar to above, this results in a total complexity of $O(d \cdot |V| \cdot \log |V|)$, which is theoretically higher than that of fixed-radius EgoNets.

To further validate this, we ran timed experiments comparing the generation of k-EgoNet subgraphs with the generation of METIS partitions on the GEOM dataset. Section 5.2 reports the average runtime of each method, computed over all graphs in the dataset.

5.3 Backbone Parameter Efficiency

Table 4 compares the number of trainable parameters across different SSL backbones. For our method, we report both the total parameters and the encoder-only count used during downstream evaluation. This distinction arises because only the target encoder is retained as the backbone, during downstream tasks, while the predictor is discarded. As a result, nearly half of the parameters are removed at this stage, allowing our method to remain competitive without increasing the parameter load for downstream evaluation, further underscoring its efficiency.

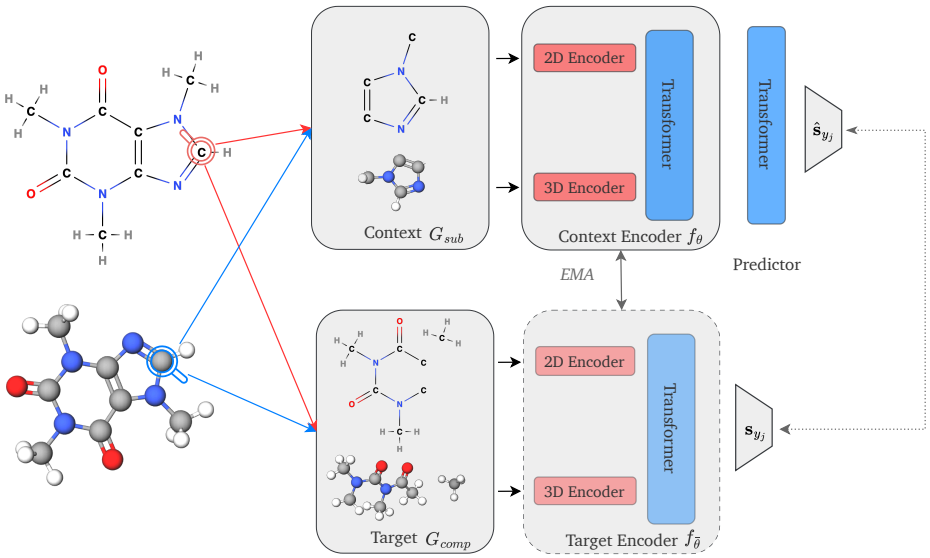

Figure 5: From each molecular graph, we sample a random node and extract its k-EgoNet [24] with $k \in \{3, 4\}$ to form complementary context and target subgraphs. Both 2D and 3D views are encoded with a GINE and a SchNet, concatenated, and passed through a transformer; the context embedding is further processed by a predictor to estimate the target. Training minimizes the mean squared $\mathcal{L}_2$ loss between predicted and encoded targets, with the target encoder updated as an exponential moving average (EMA) of the context encoder [22, 21]. For clarity, only one 3D conformation is shown, though in practice we use three.

5.4 Extension to Multi-modal data

In the following, we describe how our method can be extended to incorporate multi-modal molecular data. Specifically, for each atom $v \in V$ in the molecule, we include 3D coordinates $r_v \in \mathbb{R}^3$, taken from multiple conformers of the molecule, and use these as the 3D views of the molecule. We adapt the context encoder as follows:

Following the architecture proposed in [29], we use a message-passing neural network (MPNN) with GINE [15, 30] as the 2D encoder, and SchNet [38] as the 3D encoder used to process multiple conformers. From GINE, we obtain node-level embeddings $\{\mathbf{h}_v^{2D}\}$ for all atoms in the subgraph by averaging their intermediate representations across layers. From SchNet, we extract node-level embeddings $\{\mathbf{h}_{v,c}^{3D}\}$ for each conformer c, preserving per-atom detail across conformations. To build the multimodal sequence, we prepend a learnable classification token $\mathbf{h}_{CLS}$ and insert a learnable separation token $\mathbf{h}_{SEP}$ between the 2D and 3D components, resulting in the following multi-modal sequence:

$$\mathbf{H} = [\mathbf{h}_{CLS}, \mathbf{h}_{SEP}, \{\mathbf{h}_v^{2D}\}, \mathbf{h}_{SEP}, \{\mathbf{h}_{v,c}^{3D}\}, \mathbf{h}_{SEP}]$$

To distinguish between modalities, we add learnable modality embeddings that mark whether a token comes from the 2D or 3D graph. The full sequence is then passed through a Transformer with multiple self-attention layers to capture global dependencies both within and across modalities.

All other components remain unchanged; the full pipeline is shown in Fig. 5, and the results following the protocol described in Section 3.1 are summarized in Table 5.

5.5 Additional Details on Empirical Evaluation

For the context encoder we use a GINE [30] network containing 3 layers, with a hidden dimension of 192 and 6 Transformer layers with 8 heads and a hidden dimension of 384. The parameters are updated via backpropagation using the Adam optimizer, while the target encoder is updated through an exponential moving average (EMA) schedule, with the decay rate τ_t gradually increasing from 0.9995 to 1.0 over the course of training.

Table 5: Performance on MoleculeNet [26] with frozen backbones. **Non-CL** denotes non-contrastive and **CL** contrastive methods. We report **C-FREE$_{2D}$** (2D-only) and **C-FREE$_{MM}$** (multi-modal), each with linear probing on whole-molecule embeddings (**LIN**) or subgraph aggregation with DeepSets [28] (**DS**). Metric: ROC-AUC (↑). **Red** marks the best model and **Blue** the second best. **C-FREE** ranks first or second on 6 of 8 datasets, with **MM-LIN** best overall, while even the 2D-only variants of C-FREE outperform all baselines on average.

		MoleculeNet Datasets (Linear Probe)								
		BBBP (↑)	Tox21 (↑)	ToxCast (↑)	Sider (↑)	ClinTox (↑)	MUV (↑)	HIV (↑)	Bace (↑)	Avg (↑)
	Random Init.	$50.7_{\pm 2.5}$	$64.9_{\pm 0.5}$	$53.2_{\pm 0.3}$	$53.2_{\pm 1.1}$	$63.1_{\pm 2.3}$	$62.1_{\pm 1.3}$	$66.1_{\pm 0.7}$	$63.4_{\pm 1.8}$	59.60
CL	InfoGraph	$65.9_{\pm 0.6}$	$65.8_{\pm 0.7}$	$54.6_{\pm 0.1}$	$57.2_{\pm 1.0}$	$61.4_{\pm 4.8}$	$63.9_{\pm 1.9}$	$71.4_{\pm 0.6}$	$67.4_{\pm 4.9}$	63.44
CL	GROVER	$67.0_{\pm 0.3}$	$63.9_{\pm 0.3}$	$53.6_{\pm 0.4}$	**$59.9_{\pm 1.7}$**	$65.0_{\pm 6.4}$	$62.7_{\pm 1.4}$	$67.8_{\pm 1.0}$	$69.0_{\pm 4.7}$	63.62
CL	GraphCL	$64.7_{\pm 1.7}$	$69.1_{\pm 0.5}$	$56.2_{\pm 0.2}$	**$59.5_{\pm 0.9}$**	$60.8_{\pm 3.0}$	$60.6_{\pm 1.8}$	$72.5_{\pm 1.4}$	**$77.0_{\pm 1.7}$**	65.04
CL	JOAO	$66.1_{\pm 0.8}$	$68.1_{\pm 0.2}$	$55.1_{\pm 0.4}$	$58.3_{\pm 0.3}$	$65.3_{\pm 6.1}$	$62.4_{\pm 1.2}$	**$73.8_{\pm 1.2}$**	$71.1_{\pm 0.8}$	65.05
Non-CL	EdgePred	$54.2_{\pm 1.0}$	$66.2_{\pm 0.2}$	$54.4_{\pm 0.1}$	$56.1_{\pm 0.1}$	$65.4_{\pm 5.0}$	$59.5_{\pm 0.9}$	**$73.6_{\pm 0.4}$**	$71.4_{\pm 1.2}$	62.59
Non-CL	AttrMask	$62.7_{\pm 2.7}$	$65.7_{\pm 0.8}$	$56.1_{\pm 0.2}$	$58.3_{\pm 1.5}$	$61.9_{\pm 6.4}$	$60.9_{\pm 1.8}$	$65.5_{\pm 1.4}$	$64.8_{\pm 2.6}$	61.99
Non-CL	GPT-GNN	$62.0_{\pm 0.9}$	$64.9_{\pm 0.7}$	$55.4_{\pm 0.2}$	$55.3_{\pm 0.8}$	$55.0_{\pm 5.1}$	$61.2_{\pm 1.5}$	$71.2_{\pm 1.5}$	$61.0_{\pm 1.2}$	60.74
Non-CL	Cont. Pred	$55.5_{\pm 2.0}$	$67.9_{\pm 0.7}$	$54.0_{\pm 0.3}$	$57.1_{\pm 0.5}$	$67.4_{\pm 4.3}$	$60.5_{\pm 0.9}$	$66.2_{\pm 1.5}$	$54.4_{\pm 3.2}$	60.36
Non-CL	**C-FREE$_{2D\text{-}LIN}$**	$60.5_{\pm 1.7}$	$76.1_{\pm 0.2}$	$62.7_{\pm 0.4}$	$59.0_{\pm 0.6}$	$62.7_{\pm 1.0}$	$67.6_{\pm 0.5}$	$68.7_{\pm 0.4}$	**$75.8_{\pm 0.9}$**	66.63
Non-CL	**C-FREE$_{2D\text{-}DS}$**	$64.2_{\pm 3.8}$	**$76.7_{\pm 0.6}$**	$63.9_{\pm 0.3}$	$58.0_{\pm 0.7}$	$71.4_{\pm 3.7}$	$64.6_{\pm 3.1}$	$65.5_{\pm 0.6}$	$73.9_{\pm 0.7}$	67.27
Non-CL	**C-FREE$_{MM\text{-}LIN}$**	**$69.8_{\pm 2.6}$**	**$79.9_{\pm 1.1}$**	**$65.8_{\pm 0.7}$**	$58.5_{\pm 2.5}$	**$69.9_{\pm 1.9}$**	**$76.6_{\pm 2.8}$**	$72.8_{\pm 0.7}$	$75.3_{\pm 1.1}$	**71.07**
Non-CL	**C-FREE$_{MM\text{-}DS}$**	**$73.8_{\pm 2.1}$**	**$76.7_{\pm 0.7}$**	**$66.8_{\pm 0.2}$**	$56.4_{\pm 1.5}$	**$75.7_{\pm 2.2}$**	**$70.6_{\pm 1.0}$**	$71.9_{\pm 1.5}$	$75.5_{\pm 1.9}$	**70.92**

Table 6: Overview of tasks and sizes for the MoleculeNet datasets.

	BBBP	Tox21	ToxCast	Sider	ClinTox	HIV	Bace
# Molecules	2,039	7,831	8,575	1,427	1,478	41,127	1,513
# Tasks	1	12	617	27	2	1	1

Table 7: **C-FREE$_{SSL}$** denotes the model from checkpoint based on the best SSL loss, and **C-FREE$_{LIN}$** denotes the model from the checkpoint based on the best linear probe performance.

	MoleculeNet Datasets							
	BBBP (↑)	Tox21 (↑)	ToxCast (↑)	Sider (↑)	ClinTox (↑)	HIV (↑)	Bace (↑)	Avg (↑)
C-FREE$_{LIN}$	$60.5_{\pm 1.7}$	$76.1_{\pm 0.2}$	$62.7_{\pm 0.4}$	**$59.0_{\pm 0.6}$**	**$62.7_{\pm 1.0}$**	$68.7_{\pm 0.4}$	**$75.8_{\pm 0.9}$**	**66.6**
C-FREE$_{SSL}$	**$62.5_{\pm 2.3}$**	**$77.4_{\pm 0.4}$**	**$66.2_{\pm 0.1}$**	$52.7_{\pm 3.5}$	$52.6_{\pm 2.5}$	$69.1_{\pm 0.1}$	**$74.3_{\pm 0.5}$**	65.0

Since the EMA decay reaches $\tau_t = 1$ in the final epoch, the context and target encoders converge to identical parameters. Nevertheless, we follow [21] and report results using the target encoder.

For the choice of the scheduler we opt for a cosine scheduler without warmup. We notice that using a very small learning rate prevented convergence, while a moderate learning rate caused an early loss drop followed by stagnating representations. Adding a warmup phase allows the model to adapt gradually before the cosine decay, improving stability and representation learning. Thus we begin with a learning rate of 2×10^{-6} over 30 epochs warmup up to 5×10^{-5} and a patience of 50 epochs. For the batch size we use 256 and a weight decay of 0.04 and train for 300 epochs.

All experiments were performed on a mix of Nvidia A100/RTX 4090 GPUs and AMD EPYC 7713/Intel Xeon W-2225 CPUs for both the pre-training and downstream experiments. All experiments consumed a total of approximately 500 GPU hours, with the longest compute being consumed on the pre-training backbone run on the GEOM dataset with a total of 25 hours.

Additionally we provide an overview of the tasks and dataset sizes of the MoleculeNet dataset in Table 6.

5.6 Additional Results

Following the evaluation protocol, we select the best hyper-parameters based on average downstream performance. However, we criticize this method, as it introduces bias that is inconsistent with the self-supervised learning framework and favors downstream-specific tuning. Therefore, we also report in Table 7 results from the checkpoint achieving the best L2 loss on the self-supervised task, which we consider a more representative and principled evaluation of the model.

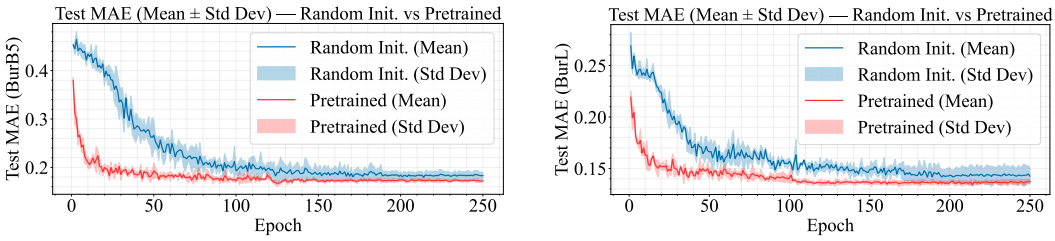

Figure 6: Best MAE on Kraken regression tasks (Sterimol BurB5 and Sterimol BurL) comparing random initialization and GEOM-pretrained models. Similarly to both targets reported in Section 3.2, pretrained models start with lower error and converge faster, while randomly initialized models fail to match their performance on average even after 250 epochs. Curves show the mean over 3 runs, with shaded regions indicating the standard deviation.

Table 8: Ablation study on the Kraken dataset (MAE ↓). We keep the encoder fixed and compare three predictors: (1) none, (2) a 2-layer MLP, and (3) a transformer. The transformer consistently achieves the best performance. The gap is especially pronounced in the linear probe (LIN. P.) setting, where the quality of the learned representations matters most. Even with full fine-tuning (FT), the no-predictor and MLP variants fail to match the transformer predictor.

	METHOD	B5 ↓	L ↓	BURB5 ↓	BURL ↓
FT	NONE	$0.381_{\pm 0.023}$	$0.494_{\pm 0.020}$	$0.202_{\pm 0.009}$	$0.157_{\pm 0.004}$
	2-LAYERS MLP	$0.315_{\pm 0.017}$	$0.396_{\pm 0.018}$	$0.185_{\pm 0.009}$	$\mathbf{0.144}_{\pm 0.004}$
	TRANSFORMER	$\mathbf{0.292}_{\pm 0.006}$	$\mathbf{0.380}_{\pm 0.023}$	$\mathbf{0.180}_{\pm 0.014}$	$0.146_{\pm 0.004}$
LIN. P.	NONE	$1.065_{\pm 0.001}$	$0.814_{\pm 0.001}$	$0.624_{\pm 0.001}$	$0.296_{\pm 0.001}$
	2-LAYERS MLP	$0.817_{\pm 0.002}$	$0.687_{\pm 0.008}$	$0.514_{\pm 0.001}$	$0.266_{\pm 0.001}$
	TRANSFORMER	$\mathbf{0.588}_{\pm 0.004}$	$\mathbf{0.554}_{\pm 0.007}$	$\mathbf{0.347}_{\pm 0.003}$	$\mathbf{0.202}_{\pm 0.008}$

Additionally, in Section 3.2 we report the results on the Sterimol BurB5 and Sterimol BurL targets from the Kraken dataset following the experiment in Section 3.2.

Finally, in Table 8, we report the explicit numbers of the probe shown in Section 3.3.

5.7 Additional Related Work

Large-Scale Supervised Pre-training An alternative to self-supervised training explores large-scale supervised pretraining on massive labeled molecular datasets, aiming to transfer knowledge to downstream tasks [**?**]. While effective in some cases, this approach still depends on labeled data and may be domain-specific, since source and target distributions can differ. In contrast, self-supervised methods avoid this reliance on labels and can transfer more flexibly. Importantly, the two strategies are complementary: self-supervised pretraining can provide strong initializations that are later fine-tuned on labeled data.

Contrastive Learning UniCorn [39] uses different molecular views and presents a unified contrastive learning framework that combines the strengths of existing methods in a single pre-training framework. 3D-Mol [40] leverages 3D conformational information by constructing hierarchical graphs and applying contrastive learning to distinguish between different molecular conformations. Additionally, GraphFP [41] leverages fragments—abstract representations of molecular substructures—to capture higher-level connectivity by introducing a contrastive task that aligns fragment embeddings with their corresponding regions in the molecular graph, enabling multi-resolution structural learning.

Latent Representation Learning CCA-SSG [42] introduces an alignment objective based on Canonical Correlation Analysis, encouraging the latent features of two augmented views to be maximally correlated while remaining de-correlated across dimensions. Complementing these augmentation-based methods, AFGRL [43] proposes a more principled strategy for view generation by identifying structurally and semantically similar anchor nodes, mitigating the reliance on handcrafted augmentations.