Substructure-Atom Cross Attention for Molecular Representation Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Designing a neural network architecture for molecular representation is crucial for AI-driven drug discovery and molecule design. In this work, we propose a new framework for molecular representation learning. Our contribution is threefold: (a) demonstrating the usefulness of incorporating substructures to node-wise features from molecules, (b) designing two branch networks consisting of a transformer and a graph neural network so that the networks fused with asymmetric attention, and (c) not requiring heuristic features and computationally-expensive information from molecules. Using 1.8 million molecules collected from ChEMBL and PubChem database, we pretrain our network to learn a general representation of molecules with minimal supervision. The experimental results show that our pretrained network achieves competitive performance on 11 downstream tasks for molecular property prediction.

1 Introduction

Predicting properties of molecules is one of the fundamental concerns in various fields. For instance, researchers apply deep neural networks (DNNs) to replace expensive real-world experiments to measure the molecular properties of a drug candidate, e.g., the capability of permeating the blood-brain barrier, solubility, and affinity. Such an attempt significantly reduces wet-lab experimentation that often takes more than ten years and costs $1 million [1, 2].

Among the DNN architectures, graph neural networks (GNNs) and Transformers are widely adopted to recognize graph structure of molecules. GNN architectures for molecular representation learning include message-passing neural network (MPNN) and directed MPNN [3, 4], where they investigate how to obtain effective node, edge, and graph representation. GNNs are powerful in capturing local information of a node, but may lack the ability to encode information from far-away nodes due to over-smoothing and over-squashing issues [5, 6]. On the other hand, Transformer-based architectures such as MAT [7] and Graphormer [8] augment the self-attention layer of a vanilla Transformer using high-order graph connectivity information. Transformers can encode global information as they consider attention between every pair of nodes from the first layer. To guide a structural bias in the attention mechanism, previous work relies on heuristic features such as the shortest path between two nodes since the naive Transformers cannot recognize the graph structure.

From the understanding of chemical structure, it is known that meaningful substructures can be found across different molecules, also known as motif or fragments [9]. For example, carbon rings and NO2 groups are typical substructures contributed to mutagenicity [10] showing that proper usage of substructures can help a property prediction. Molecular substructures are often represented as

Submitted to NeurIPS 2022 AI for Science Workshop.

(a) Indistinguishable molecules by traditional GNNs (b) Indistinguishable molecules by MACCS keys

Figure 1: Traditional GNNs, such as GCN and GAT, cannot distinguish the two molecular graphs in (a). However, it can be easily distinguished through simple 6-ring and 5-ring substructure information. On the other hand, the two molecules in (b) have similar substructures, so atom features from neighborhood are necessary to discriminate the two molecules.

molecular fingerprints or molecular fragmentation. Molecular fingerprints such as MACCS (Molecular ACCess System) keys [11] and Extended-Connectivity Fingerprints (ECFPs) [12] represent a molecule into a fixed binary vector where each bit indicates the presence of a certain motif in the molecule. With a predefined fragmentation dictionary, such as BRICS [13] or tree decomposition [14], a molecule can be decomposed into distinct partitions. Interestingly, machine learning algorithms that utilize molecular substructures still show competitive performance on some datasets to deep learning models [15, 7].

We propose a fusion architecture between a GNN and Transformer to incorporate molecular graph information and molecular substructures. Molecular substructures and graph are encoded through Transformer and GNN, respectively. The Transformer is designed to recognize the molecular substructures, where the substructure information is mixed through self-attention to obtain better representations. With the Transformer only architecture, however, local information of molecules, such as atoms, bonds, and connectivity, can be lost from the structures. For example, the two molecules shown in Figure 1b share the same representation with MACCS keys while having different structures. To overcome, we use a separate GNN branch for preserving local information. In our model, we inject the GNN feature into the intermediate Transformer layers through the fusing network. In this way, substructures and local node information are interactively fused, producing a final representation for molecular graphs.

We name our network as ***Substructure-Atom Cross Attention (SACA)*** as it uses substructure as well as atom information in molecules and fuses them through cross-attention. The architectural choices allows us to avoid heuristic features such as the shortest-path or 3-dimensional distance in attention layers, which are required in many existing Transformer architectures for molecular representation learning [7, 8]. Furthermore, our model reduces the complexity for attention calculation over the node-level Transformer models from $O(N^2)$ to $O(N)$, where N is the number of atoms. To demonstrate the empirical effectiveness of the proposed network and see the ability to capture the general representation of molecules, we evaluate our model on 11 downstream tasks from MoleculeNet [16]. Our approach achieves the competitive performance on 11 downstream tasks.

In what follows, we summarize the key contributions and benefits.

- We propose a novel network that combines the information from substructures and node features in a molecule. Our model combines the advantages of both Transformer and GNN architectures to represent the information given to each architecture.

- We show the effectiveness of our model for molecular representation learning. Our model achieves competitive performance upon strong baseline models on 11 molecular property tasks.

- Our model does not require computationally-expensive heuristic information of molecular graphs.

2 Related Work

2.1 Architectures for Molecular representation learning

Graph neural networks (GNNs) Most common architecture for molecular representation learning is the GNNs since molecules can be naturally represented as a graph structure; a node as an atom, an edge as a connection. Researchers have actively investigated variations of GNN architectures [3, 4, 17, 18], for molecules. For example, MPNN [3] generalizes the message passing frameworks and explores some variants that predict molecular properties. Directed MPNN (DMPNN) [4] proposed to replace the node-based message by edge-based messages to avoid unnecessary message loops. Next, communicative MPNN (CMPNN) [18] improved DMPNN by additionally considering the node-edge interaction during the message passing phase. AttentiveFP [17] extends the graph attention mechanism to allow for nonlocal effects at the intramolecular level.

Despite the advance of GNN architectures for molecular representation learning, there are known problems in GNN, such as over-smoothing and over-squashing problems [5, 6], which means the node representations become too similar, and the information from far nodes does not propagate well as the number of neighbors increases exponentially. Furthermore, [19] analyzes expressive powers of standard GNNs following neighborhood aggregation scheme and shows that they are bounded to Weisfeiler-Lehman test (WL-test) [20]. Therefore, GNNs with standard message passing cannot learn to discern a simple substructure such as cycles. For example, the two molecules in 1a cannot be distinguished by WL-test, hence standard GNNs cannot distinguish these two molecules [21]. A solution to this limited representation power of GNNs is to directly incorporate important substructures in the representation learning framework.

Transformers With recent advance of Transformer architectures and their promising performance in various domains, including NLP and computer vision [22, 23], Transformer-based architectures for molecular representation learning [7, 8, 24, 25] have been developed. Transformer architecture calculates pair-wise attention between every node from the first layer. Therefore, it can effectively capture the global information of a graph. However, a vanilla Transformer architecture [26] is not directly applicable for molecular graph representation because it cannot incorporate structural information such as the edge and connectivity in graphs. To bridge the gap, advanced Transformer architectures alter the self-attention layer [7, 8, 25] or incorporate message passing networks into Transformer architectures for input feature [24]. Specifically, MAT [7] uses adjacency and distance matrix of atoms to augment the self-attention layer. GROVER [24] runs Dynamic Message Passing Network (dyMPN) over the input node and edge features to extract queries, keys and values for self-attention layers. CoMPT [27] uses the shortest path information between two nodes, and Graphormer [8] encodes node's degree, edges and the shortest path with edges between two nodes for molecular data. Although these graph-specific features such as the shortest path or 3D distance are found to be useful in graph representation, these features can be heuristic and impose excessive computational overhead, which limits the model's applicability to large molecules such as proteins. Our proposed model does not require any computationally expensive heuristic features such as 3D information or shortest path for preprocessing or computation in attention layer.

2.2 Molecular Substructure

Substructure information extracted from molecules has been widely used in molecular generation, property prediction, and virtual screening [28, 29, 30, 14]. ECFPs [12] encode existing substructures within a circular distance from each atom in a molecule. PMTNN [31] is a multi-task network that takes ECFPs as an input to predict molecular properties on various datasets. MACCS keys [11] extract molecules substructure depending on the presence of pre-defined functional groups. One example of the usage of MACCS keys is to encode known ligands of each protein, which leads to an improvement of prediction performance in protein-ligand interaction [32]. Molecular fragmentation method such as BRICS fragmentation [13, 33] or tree decomposition [14] is to decompose molecules in non-overlapping partitions with pre-defined rules. BRICS fragmentation divides molecules following chemical reaction based rules. Tree decomposition proposed in [14] also extracts junction tree

by contracting certain edges. One node in the junction tree represents a substructure of original molecules. Our model receives molecular substructures as input to supplement GNN's expressivity and it can flexibly encode any substructure vocabulary.

Our model combines Transformer and GNN, with molecular substructures and molecular graph as inputs to each network. There are existing studies that have also considered combining GNN and Transformer architecture [34, 35, 36] to learn graph representations. Specifically, DMP [34] utilizes GNN and Transformer to encode graphs and SMILES representation of molecules and train both branches using a consistency loss to match the two outputs of input molecules. PoseGTAC [35] and GraphFormers [36] combine GNN and Transformer layer alternatively to enlarge the receptive field or to mix the output of Transformer. Our model is the first to encode substructures through Transformer and inject atom features through a separate GNN. In this way, we preserve both substructures and local atom features of molecules.

3 Model

In this section, we explain the architecture of our model.

Overall architecture Figure 2 shows the overall architecture of our model. Our network consists of two branches: (i) a Transformer branch that uses the molecular substructures as input and (ii) a GNN branch that uses a molecular graph as input. The two branches have different roles. First, the Transformer branch is intended to capture global information of molecules. It receives the molecular substructures that have important role in molecular properties, but cannot be easily captured by GNNs, and learn the overall representation of molecules. On the other hand, the GNN branch is intended to capture local node information of molecules. The two different levels of information are mixed through the fusing network in the Transformer branch.

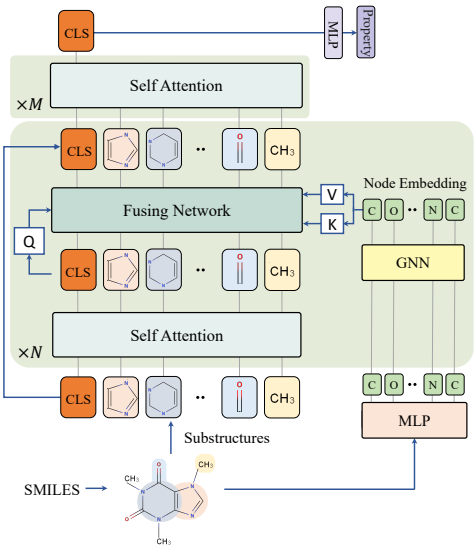


Figure 2: Overall architecture. Our model consists of Transformer and GNN branches. Transformer encodes molecular substructures, while GNN encodes atomic information. The self-attention and fusing network are alternatively stacked N times, followed by M more self-attention layer to refine the substructure feature. The final CLS token is used to predict the molecular properties.

Transformer branch Our Transformer branch is to incorporate both molecular substructure information and local node features. The input token for Transformer is the substructure embeddings of molecules. Predefined substructures are first detected and then projected into separate embedding vectors. For example, in Figure 2, substructures such as N-Heterocycle, carbon-oxygen bond and methyl group are detected and embedded into learnable embedding vectors. To identify substructure from the input molecules, we use MACCS keys [11], which indicates the presence of motifs in a molecule. Note that our architecture is not limited to certain molecular substructures, but it can flexibly receive any substructure vocabulary. The embeddings of substructures are mixed together and refined as they are passed through the self-attention module.

The substructure embeddings after self-attention layer are fused with node embeddings from a separate GNN branch. The fusing network computes cross-attention between substructures and nodes where substructures are used as query and nodes are used as key and value. The detailed computation of the fusing network is shown in Figure 3. In the fusing network, the cross-attention between each pair of substructure embedding and node embedding is computed.

To be specific, for a given molecule having n atoms and m extracted substructures, we have substructure embeddings $E_s \in \mathbb{R}^{m \times d}$ and node embeddings $E_n \in \mathbb{R}^{n \times d}$ where d is the embedding dimension. Then, the cross-attention is computed as follows:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{(E_s W_Q)(E_n W_K)^T}{\sqrt{d_k}}\right)(E_n W_V), \tag{1}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d \times d_k}$ are learnable parameters. The cross-attention module outputs $E'_s \in \mathbb{R}^{m \times d_k}$. Through this fusing network, the substructure embeddings aggregate the local information from node embeddings. Structurally important nodes are aggregated with more weights. Instead of designing heuristic weights on the nodes, our model can learn to select structurally important nodes related to graph-level property by cross-attention. Additionally, as the attention is computed between substructures and nodes, the space and time complexity of the self and cross attention map of our model is linear to the number of atoms, i.e., $O(N)$, whereas other transformer architectures for molecular graphs have quadratic complexity.

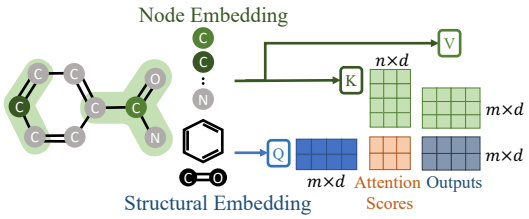


Figure 3: Illustration of structural and node embedding for cross attention computation. Cross attention is computed with the illustrated query, key and value.

The self-attention between substructures and fusing network with node embeddings are repeated iteratively N times. Before making the final prediction, we add M self-attention layers at the end of the network for making refinement on the substructures. We add a residual connection from the input tokens to every output of the fusing network. This ensures the input structural information last throughout the entire network. Furthermore, we use a CLS token similar to special classification token in BERT [22] to aggregate the global representation of molecules. The CLS token is shown in Figure 2. It is a learnable latent vector and passed to the Transformer branch attached to the substructure tokens as input. A CLS token has been used in many Transformer-based architectures [7, 8] for molecular representation learning. We attach an MLP head to the final CLS token to perform graph-level prediction tasks.

GNN branch The GNN branch is used to extract local node features from molecular graphs. For GNN architecture, we use GIN [19] with jumping knowledge [37]. The computed node features are injected into the Transformer through the fusing network with the same hierarchy. For example, 0-hop node representations are injected into the first cross-attention module and 1-hop representations for the second cross-attention module. This allows the model to encode local node features progressively from the shallow to deeper layers.

The computation of node representation through GNNs and injection to the Transformer allow us to take advantage of both GNNs and Transformer architectures. Substructures that are hard to be captured by GNNs, but essential to molecular properties, are first detected and encoded through Transformer. Meanwhile, local node information that can be lost in using substructures alone is effectively captured by GNNs and fused with substructures. Additionally, our architecture does not require computationally expensive high-order graph-level information. As Transformer cannot naturally incorporate a graph's edge connectivity information, existing work [24, 8, 27] mainly focuses on how to add structural bias such as the shortest path or 3D distance between two nodes into the Transformer self-attention computation. However, these structural biases require computationally expensive preprocessing of the molecular datasets. Our network that utilizes GNN as a separate branch can avoid these limitations.

4 Experiment

4.1 Experimental Setting

Pretraining We pretrain our network to obtain molecular representation transferable to various molecular datasets and tasks. For pretraining, we extracted 200 real-valued descriptors of physico-chemical properties from the pretraining datasets using RDKit [38] and train our network to predict these properties. As the 200 molecular descriptors include a diverse set of molecular properties, the model can learn a representation of molecules that can be used for various downstream tasks.

Dataset We collected 1,858,081 number of unlabeled molecules from ChEMBL and PubChem databases [39, 40]. ChEMBL and PubChem are large-scale databases that include a variety of chemical and physical properties, and biological activities of molecules. To obtain molecular substructures, we utilize MACCS key [11], a 166 dimensional vector that indicates a presence of certain substructure in molecules, and extract this for every molecule using RDKit [38]. We use OGB package [15] to convert SMILES [41], a text-representation for molecules, to molecular graphs.

Implementation details We set $M = 4$ and $N = 3$, where M and N are defined in section 3. We set 768-dimensional hidden units and 16 attention heads. When pretraining, we used the AdamW optimizer [42] with a learning rate of 1e-4. We divide the pretraining dataset into a 9:1 ratio and use them for training and validation sets. The model is trained for 10 epochs, and the model with the best validation loss is used for downstream tasks. Further details for pretraining setting is presented in Appendix D.

4.2 Downstream tasks

Tasks We evaluate the performance on six classification tasks (BBBP, SIDER, ClinTox, BACE, Tox21, and ToxCast) and five regression tasks (FreeSolv, ESOL, Lipo, QM7 and QM8) from MoleculeNet [16]. The dataset statistics are summarized in Table 4. Each task is related to molecular property from low-level, for example, water solubility in ESOL to high-level, possibility of blood-brain barrier penetraion in BBBP. Further details about the downstream tasks are available in Appendix B.

Experimental setting To evaluate each model, we use 3 different scaffold splits [16] following [24]. Scaffold split divides structurally different molecules into different subsets and provides more challenging and realistic test environment. From the pretrained model, we replace the last MLP layer of the network with the task-specific MLP heads. For each downstream dataset, we train our model for 100 epochs and report the test score corresponding to the best validation epoch. We tune hyperparameters with Bayesian optimization search with a budget of 100 for learning rate, dropout, weight decay and the number of last prediction heads. The hyperparameter search range is provided in Appendix D.

We compare the performance of our model on the downstream tasks with several GNN and Transformer based state-of-the-arts approaches for molecule representation learning. TF-Robust [31] is a DNN-based model that takes molecular fingerprints. GNN-based models include GraphConv [43], Weave [44] and SchNet [45] which are 3 graph convolutional networks, MPNN [3], DMPNN [4], MGCN [46] and CMPNN [18] which are GNN models considering the edge features during message passing. AttentiveFP [17] is an extension of graph attention network for molecule representation. Transformer-based models include GROVER [24], MAT [7], Graphormer [8] and CoMPT [27]. Among the baselines, N-GRAM [47], [48], GraphLoG [49], MAT, GROVER, Graphormer, GEM [50] and MPG [51] are models that use pretraining strategies. We report GROVER base model for a fair comparison in terms of the number of parameters. To reproduce the results for models using pretraining strategies, we use the pretrained model made available by the authors. We reproduce the results of MPG due to the different splits used in [51][1]. We also evaluate our model with the same data split used for MPG and the results can be found in the appendix.

[1]Note that there is a mismatch between the splits used in the original paper of MPG and the one used in the author's repository. For both cases, our model performs better than MPG under the same splits. The additional experiments are available in Appendix G.

Table 1: Comparison on small-scale datasets. We report the average and standard deviation (in brackets) over three splits. We mark the best and the second-best performances in **bold yellow** and light yellow, respectively. The baseline results except for MAT, Graphormer, CMPNN, CoMPT, GraphLoG, GEM and MPG are taken from [24].

Classification Tasks

Method	Pre.	BBBP ↑	SIDER ↑	ClinTox ↑	BACE ↑	Tox21 ↑	ToxCast ↑	Rank
TF_Robust [31]	-	.860(.087)	.607(.033)	.765(.085)	.824(.022)	.698(.012)	.585(.031)	15.0
Weave [44]	-	.837(.065)	.543(.034)	.823(.023)	.791(.008)	.741(.044)	.678(.024)	16.0
GraphConv [43]	-	.877(.036)	.593(.035)	.845(.051)	.854(.011)	.772(.041)	.650(.025)	13.2
SchNet [45]	-	.847(.024)	.545(.038)	.717(.042)	.750(.033)	.767(.025)	.679(.021)	16.2
MPNN [3]	-	.913(.041)	.595(.030)	.879(.054)	.815(.044)	.808(.024)	.691(.013)	11.0
DMPNN [4]	-	.919(.030)	.632(.023)	.897(.040)	.852(.053)	.826(.023)	.718(.011)	5.7
MGCN [46]	-	.850(.064)	.552(.018)	.634(.042)	.734(.030)	.707(.016)	.663(.009)	17.0
AttentiveFP [17]	-	.908(.050)	.605(.060)	.933(.020)	.863(.015)	.807(.020)	.579(.001)	10.0
CMPNN [18]	-	**.940**(.009)	.612(.006)	.931(.003)	.868(.033)	.805(.017)	.722(.005)	5.0
CoMPT [27]	-	.930(.019)	.605(.011)	.818(.081)	.851(.043)	.790(.031)	.716(.010)	9.5
N-GRAM [47]	✓	.912(.013)	.632(.005)	.855(.037)	.876(.035)	.769(.027)	-	8.4
Hu. et.al [48]	✓	.915(.040)	.614(.006)	.762(.058)	.851(.027)	.811(.015)	.714(.019)	9.2
MAT [7]	✓	.922(.035)	.617(.012)	.853(.079)	.830(.045)	.810(.015)	.712(.004)	8.5
GROVER [24]	✓	.936(.008)	**.656**(.006)	.925(.013)	**.878**(.016)	.819(.020)	.723(.010)	2.3
Graphormer [8]	✓	.938(.032)	.625(.009)	.913(.056)	.848(.023)	.801(.013)	.718(.007)	6.7
GraphLoG [49]	✓	.913(.024)	.595(.039)	-	.845(.012)	.773(.010)	.677(.008)	13.0
MPG* [51]	✓	.922(.039)	.628(.014)	-	.864(.028)	.800(.024)	.712(.009)	7.4
GEM [50]	✓	.921(.026)	.603(.012)	-	.872(.036)	.815(.016)	.720(.010)	6.6
Ours	✓	.934(.018)	.646(.009)	**.935**(.014)	.877(.032)	**.829**(.013)	**.730**(.005)	**1.8**

Regression Tasks

Method	Pre.	FreeSolv ↓	ESOL ↓	Lipo ↓	QM7 ↓	QM8 ↓	Rank
TF_Robust [31]	-	4.122(.085)	1.722(.038)	.909(.060)	120.6(9.6)	.024(.001)	15.2
Weave [44]	-	2.398(.250)	1.158(.055)	.813(.042)	94.7(2.7)	.022(.001)	11.8
GraphConv [43]	-	2.900(.135)	1.068(.050)	.712(.049)	118.9(20.2)	.021(.001)	12.2
SchNet [45]	-	3.215(.755)	1.045(.064)	.909(.098)	74.2(6.0)	.020(.002)	11.4
MPNN [3]	-	2.185(.952)	1.167(.430)	.672(.051)	113.0(17.2)	.015(.002)	10.0
DMPNN [4]	-	2.177(.914)	.980(.258)	.653(.046)	105.8(13.2)	.0143(.002)	7.8
MGCN [46]	-	3.349(.097)	1.266(.147)	1.113(.041)	77.6(4.7)	.022(.002)	13.6
AttentiveFP [17]	-	2.030(.420)	.853(.060)	.650(.030)	126.7(4.0)	.0282(.001)	8.8
CMPNN [18]	-	2.254(.356)	.841(.090)	.606(.040)	70.6(2.7)	.0136(.001)	4.8
CoMPT [27]	-	2.125(.590)	.898(.053)	.632(.038)	65.3(3.4)	.0145(.002)	5.6
N-GRAM [47]	✓	2.512(.190)	1.100(.160)	.876(.033)	125.6(1.5)	.0320(.003)	14.0
MAT [7]	✓	2.116(.152)	.833(.122)	.668(.025)	93.6(13.8)	.0178(.002)	6.6
GROVER [24]	✓	**1.592**(.072)	.888(.116)	**.563**(.030)	72.5(5.9)	.0172(.002)	4.4
Graphormer [8]	✓	2.089(.150)	.827(.086)	.674(.035)	171.3(10.7)	.0140(.002)	7.2
MPG* [51]	✓	2.44(.520)	.926(.159)	.682(.013)	-	-	10.7
GEM [50]	✓	2.21(.374)	.885(.115)	.617(.023)	**58.9**(3.9)	.0135(.001)	4.4
Ours	✓	1.750(.170)	**.822**(.073)	.575(.009)	63.5(3.4)	**.0134**(.002)	**1.6**

Results Table 1 shows the overall results of the baselines and our model on 11 MoleculeNet datasets. Our model achieved the best performance on five downstream tasks: ClinTox, Tox21, ToxCast, ESOL and QM8, and the second best performance on five downstream tasks: SIDER, BACE, FreeSolv, Lipo and QM7. We also computed the average rank for the classification and regression tasks, separately. Our model achieved the best average rank among all compared models. The result shows that our model generalizes well across different downstream tasks, which means local information aggregated from GNN can propagate globally as interacting with structural information in Transformer. We find that some substructures weight high attention to nodes consisting of the substructures, and CLS token used for prediction also focuses on the substructures from attention scores in self-attention.

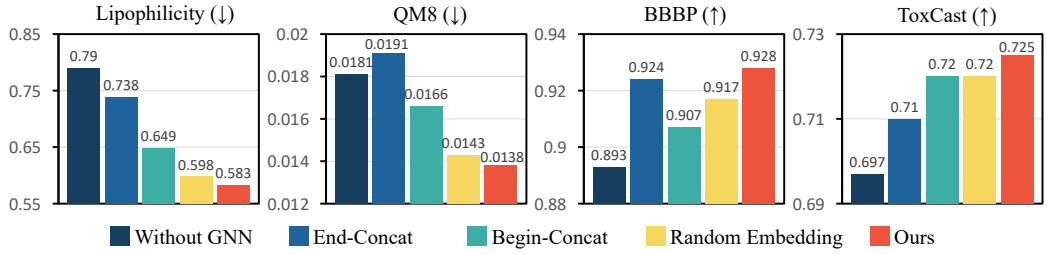


Figure 4: Ablation study. Four variations of our model (Without GNN, End-Concat, Begin-Concat, Random Embedding) on four downstream datasets.

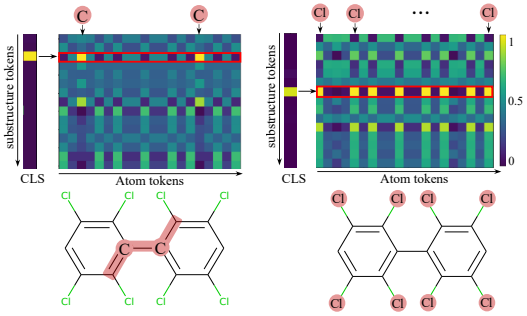
4.3 Ablation study

We report ablation study to justify each component and flexibility of our model architecture. For the ablation study, we reduce the hyperparameter search space to only 6 learning rates {1e-3, 5e-4, 1e-4, 5e-5, 1e-5, 5e-6} to facilitate the comparison between different models. We report the test scores based on the hyperparameters from the best validation scores.

Ablation on model components Figure 4 shows the performance comparison between four variations of our model on four downstream datasets. We first verify the performance of our model without GNN branch. To do that, we replace all cross-attention layers with self-attention layers and exclude GNN branch. Begin-Concat and End-Concat examine different ways of combining substructure and local features. Begin-Concat runs a vanilla-transformer encoder on top of a concatenated atom and substructure embeddings. End-Concat runs a vanilla-transformer encoder on top of atom embeddings and concatenates the substructure feature at the end to make the final prediction. Random Embedding examines the effect of substructural embeddings by replacing them with random learnable embeddings. Figure 4 shows our model outperforms other variations on all datasets, which justifies the necessity of each component.

4.4 Analysis

Attention map Figure 5 shows the attention weights in a cross-attention and a self-attention layer. In the cross-attention layer, each row and column correspond to substructure and node in the given molecule, so we can interpret the attention score as the degree of focus between the substructures and nodes. We check where the CLS token gives more attention since it is used for prediction. We find out the CLS token gives strong attention to specific substructures. More interestingly, often a substructure gives more attention on the nodes that consists the substructure itself. Despite of not having any structural information between input substructures and nodes as input, our model can identify structurally related nodes in the cross attention layer, showcasing the ability of understanding molecular structure.



(a) Ring-Chain-Ring bond (b) Chlorine atom

Figure 5: Attention visualization. CLS token strongly attends to Ring-Chain-Ring bond and Cl substructures of the input molecule. The cross attention maps show the substructures capture the atoms related to the substructure.

5 Conclusion

In this paper, we propose a novel framework that incorporates Transformer and GNN architecture for molecular representation learning. Our model takes advantages of the two architectures to aggregate substructure and local information. With the cross attention mechanism in fusing network, our model could achieve state-of-the-art performance on various molecular property prediction benchmarks. Overall, our work highlights the effectiveness of SACA for molecular representation learning.

References

[1] James P Hughes, Stephen Rees, S Barrett Kalindjian, and Karen L Philpott. Principles of early drug discovery. *British journal of pharmacology*, 162(6):1239–1249, 2011.

[2] Richard C Mohs and Nigel H Greig. Drug discovery and development: Role of basic biological research. *Alzheimer's & Dementia: Translational Research & Clinical Interventions*, 3(4):651–657, 2017.

[3] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[4] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, et al. Analyzing learned molecular representations for property prediction. *Journal of chemical information and modeling*, 2019.

[5] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on artificial intelligence*, 2018.

[6] Uri Alon and Eran Yahav. On the Bottleneck of Graph Neural Networks and its Practical Implications. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[7] Łukasz Maziarka, Tomasz Danel, Sławomir Mucha, Krzysztof Rataj, Jacek Tabor, and Stanisław Jastrzębski. Molecule attention transformer. *arXiv preprint arXiv:2002.08264*, 2020.

[8] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do Transformers Really Perform Bad for Graph Representation? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[9] Christopher W Murray and David C Rees. The rise of fragment-based drug discovery. *Nature chemistry*, 2009.

[10] Asim Kumar Debnath, Rosa L Lopez de Compadre, Gargi Debnath, Alan J Shusterman, and Corwin Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry*, 34(2):786–797, 1991.

[11] Joseph L Durant, Burton A Leland, Douglas R Henry, and James G Nourse. Reoptimization of MDL keys for use in drug discovery. *Journal of chemical information and computer sciences*, 2002.

[12] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 2010.

[13] Jörg Degen, Christof Wegscheid-Gerlach, Andrea Zaliani, and Matthias Rarey. On the art of compiling and using'drug-like'chemical fragment spaces. *ChemMedChem: Chemistry Enabling Drug Discovery*, 3(10):1503–1507, 2008.

[14] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[15] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

[16] 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*, 2018.

[17] Zhaoping Xiong, Dingyan Wang, Xiaohong Liu, Feisheng Zhong, Xiaozhe Wan, Xutong Li, Zhaojun Li, Xiaomin Luo, Kaixian Chen, Hualiang Jiang, et al. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. *Journal of medicinal chemistry*, 63(16):8749–8760, 2019.

[18] Ying Song, Shuangjia Zheng, Zhangming Niu, Zhang-Hua Fu, Yutong Lu, and Yuedong Yang. Communicative representation learning on attributed molecular graphs. In *IJCAI*, 2020.

[19] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[20] 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.

[21] Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Lio, Guido F Montufar, and Michael Bronstein. Weisfeiler and lehman go cellular: Cw networks. *Advances in Neural Information Processing Systems*, 34:2625–2640, 2021.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *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)*, 2019.

[23] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[24] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. GROVER: Self-supervised Message Passing Transformer on Large-scale Molecular Data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[25] Łukasz Maziarka, Dawid Majchrowski, Tomasz Danel, Piotr Gaiński, Jacek Tabor, Igor Podolak, Paweł Morkisz, and Stanisław Jastrzębski. Relative Molecule Self-Attention Transformer. *arXiv preprint arXiv:2110.05841*, 2021.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[27] Jianwen Chen, Shuangjia Zheng, Ying Song, Jiahua Rao, and Yuedong Yang. Learning attributed graph representations with communicative message passing transformer. In *IJCAI*, 2021.

[28] Peter Willett, John M Barnard, and Geoffrey M Downs. Chemical similarity searching. *Journal of chemical information and computer sciences*, 38(6):983–996, 1998.

[29] Robert D Brown and Yvonne C Martin. Use of structure- activity data to compare structure-based clustering methods and descriptors for use in compound selection. *Journal of Chemical Information and Computer Sciences*, 36(3):572–584, 1996.

[30] Hanna Eckert and Jürgen Bajorath. Molecular similarity analysis in virtual screening: foundations, limitations and novel approaches. *Drug discovery today*, 12(5-6):225–233, 2007.

[31] Bharath Ramsundar, Steven Kearnes, Patrick Riley, Dale Webster, David Konerding, and Vijay Pande. Massively multitask networks for drug discovery. *arXiv preprint arXiv:1502.02072*, 2015.

[32] Li Li, Ching Chiek Koh, Daniel Reker, JB Brown, Haishuai Wang, Nicholas Keone Lee, Hien-haw Liow, Hao Dai, Huai-Meng Fan, Luonan Chen, et al. Predicting protein-ligand interactions based on bow-pharmacological space and bayesian additive regression trees. *Scientific reports*, 9(1):1–12, 2019.

[33] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Chee-Kong Lee. Motif-based graph self-supervised learning for molecular property prediction. *Advances in Neural Information Processing Systems*, 34:15870–15882, 2021.

[34] Jinhua Zhu, Yingce Xia, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. Dual-view molecule pre-training. *arXiv preprint arXiv:2106.10234*, 2021.

[35] Yiran Zhu, Xing Xu, Fumin Shen, Yanli Ji, Lianli Gao, and Heng Tao Shen. Posegtac: Graph transformer encoder-decoder with atrous convolution for 3d human pose estimation. In *IJCAI*, pages 1359–1365, 2021.

[36] Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. Graphformers: Gnn-nested transformers for representation learning on textual graph. *Advances in Neural Information Processing Systems*, 34:28798–28810, 2021.

[37] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, pages 5453–5462. PMLR, 2018.

[38] Greg Landrum. Rdkit: Open-source cheminformatics software. 2016.

[39] Sunghwan Kim, Paul A Thiessen, Evan E Bolton, Jie Chen, Gang Fu, Asta Gindulyte, Lianyi Han, Jane He, Siqian He, Benjamin A Shoemaker, et al. PubChem substance and compound databases. *Nucleic acids research*, 2016.

[40] Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, et al. Chembl: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 2012.

[41] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 1988.

[42] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[43] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[44] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, pages 595–608, 2016.

[45] Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in neural information processing systems*, 30, 2017.

[46] Chengqiang Lu, Qi Liu, Chao Wang, Zhenya Huang, Peize Lin, and Lixin He. Molecular property prediction: A multilevel quantum interactions modeling perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1052–1060, 2019.

[47] Shengchao Liu, Mehmet F Demirel, and Yingyu Liang. N-gram graph: Simple unsupervised representation for graphs, with applications to molecules. *Advances in neural information processing systems*, 32, 2019.

[48] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for Pre-training Graph Neural Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[49] Minghao Xu, Hang Wang, Bingbing Ni, Hongyu Guo, and Jian Tang. Self-supervised graph-level representation learning with local and global structure. In *International Conference on Machine Learning*, pages 11548–11558. PMLR, 2021.

[50] Xiaomin Fang, Lihang Liu, Jieqiong Lei, Donglong He, Shanzhuo Zhang, Jingbo Zhou, Fan Wang, Hua Wu, and Haifeng Wang. Geometry-enhanced molecular representation learning for property prediction. *Nature Machine Intelligence*, 4(2):127–134, 2022.

[51] Pengyong Li, Jun Wang, Yixuan Qiao, Hao Chen, Yihuan Yu, Xiaojun Yao, Peng Gao, Guotong Xie, and Sen Song. An effective self-supervised framework for learning expressive molecular global representations to drug discovery. *Briefings in Bioinformatics*, 22(6):bbab109, 2021.

[52] Benedek Fabian, Thomas Edlich, Héléna Gaspar, Marwin Segler, Joshua Meyers, Marco Fiscato, and Mohamed Ahmed. Molecular representation learning with language models and domain-relevant auxiliary tasks. *arXiv preprint arXiv:2011.13230*, 2020.

[53] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[54] Tomasz Danel, Przemysław Spurek, Jacek Tabor, Marek Śmieja, Łukasz Struski, Agnieszka Słowik, and Łukasz Maziarka. Spatial graph convolutional networks. In *International Conference on Neural Information Processing*, pages 668–675. Springer, 2020.

A Node and Edge Features

In this section, we present the node and edge features of molecules used for GNN branch. We used OGB package [15] to convert SMILES strings [41] to molecular graphs. The molecular graphs are encoded through GIN [19] and injected into the Transformer branch by the cross-attention. The molecular graphs have the following node and edge features.

Node features. Each node has the following 9 dimensional features as shown in Table 2.

Table 2: Node Features.

Index	Description	Range
0	Atomic num	[1, 118], other
1	Chirality	unspecified, tetrahedral cw, tetrahedral ccw, other
2	Degree	[0, 10], other
3	Formal Charge	[-5, 5], other
4	Num Hydrogen	[0, 8], other
5	Num Radical Electron	[0, 4], other
6	Hybridization	SP, SP2, SP3, SP3D, SP3D2, other
7	Is Aromatic	False, True
8	Is in Ring	False, True

Edge features. Each edge has the following 3 dimensional features as shown in Table 3.

Table 3: Edge Features.

Index	Description	Range
0	Bond Type	single, double, triple, aromatic, other
1	Bond Stereo	stereonone, stereoz, stereoe, stereocis, stereotrans, stereoany
2	Is Conjugated	False, True

B Details of Downstream Datasets

We used 11 binary graph classification and regression datasets: BBBP, SIDER, ClinTox, BACE, Tox21, ToxCast, FreeSolv, ESOL, Lipophilicity, QM7 and QM8 from Moleculenet [16]. The statistics of each dataset is shown in Table 4. The details of each dataset are shown in Table 5. Through the various datasets, we can test the generalization ability of our pretrained model.

Table 4: Statistics of eleven datasets from MoleculeNet [16] used for downstream tasks.

(a) Classification tasks

Dataset	Size	# Tasks	Metric
BBBP	2,039	1	ROC-AUC
SIDER	1,427	27	ROC-AUC
ClinTox	1,478	2	ROC-AUC
BACE	1,513	1	ROC-AUC
Tox21	7,831	12	ROC-AUC
ToxCast	8,575	617	ROC-AUC

(b) Regression tasks

Dataset	Size	# Tasks	Metric
FreeSolv	642	1	RMSE
ESOL	1,128	1	RMSE
Lipophilicity	4,200	1	RMSE
QM7	6,830	1	MAE
QM8	21,786	12	MAE

Table 5: Detailed description for each downstream dataset.

Dataset	Description
BBBP	Binary classification task to predict a molecule's blood-brain barrier penetration ability
SIDER	Marketed drugs with its adverse drug reactions
ClinTox	Qualitative data of drugs approved by the FDA and those that have failed clinical trials for toxicity reasons
BACE	Binary classification task to predict a molecule's binding result for a set of inhibitors of human β-secretase 1
Tox21	Qualitative toxicity measurements on 12 biological targets
ToxCast	Toxicology data for a large library of compounds based on in vitro high-throughput screening, including experiments on over 600 tasks
FreeSolv	Regression task to predict hydration free energy of small molecules in water
ESOL	Regression task to predict water solubility in terms of log solubility in mols per litre
Lipophilicity	Experimental results of octanol/water distribution coefficient
QM7	A subset of GDB-13 composed of all molecules of up to 23 atoms (including 7 heavy atoms C, N, O, and S), totalling 7165 molecules
QM8	Computer-generated quantum mechanical properties

C Molecular Structural Keys

In this section, we explain the details of molecular substructures that we used for our model, i.e., Molecular ACCess System (MACCS) keys [11]. We extract MACCS keys for each molecule using RDKit package [38]. MACCS keys have 166-dimensional features where each binary label indicates the presence of a particular substructure in the given molecule. For example, the 139th index of MACCS keys indicates the presence of the Hydroxy group (-OH), and the 162nd index indicates the presence of the aromatic ring in a molecule. The full list of 166 MACCS keys can be found in the document [2].

Figure 6 shows examples of how MACCS keys encode substructure information into a binary bit string. The canonical SMILES representations of the left and right molecules are C1=CC=C(C=C1)O and Cc1cc(Cl)cc(Cl)c1CNC(=O)c1cccs1, respectively.

Table 6 shows the MACCS keys statistics for pretraining and downstream datasets. It includes the average number of MACCS keys that molecules in each dataset contain and the standard deviation. Also, the minimum and the maximum number of MACCS keys for each dataset are reported.

D Experimental Setting

In this section, we explain further details of our experimental setting.

Pretraining. The default hyperparameters for pretraining are listed in Table 7. All attention layers have 16 attention heads and 768 hidden dimension. Our model has 41M parameters.

[2] `https://github.com/rdkit/rdkit/blob/master/rdkit/Chem/MACCSkeys.py`

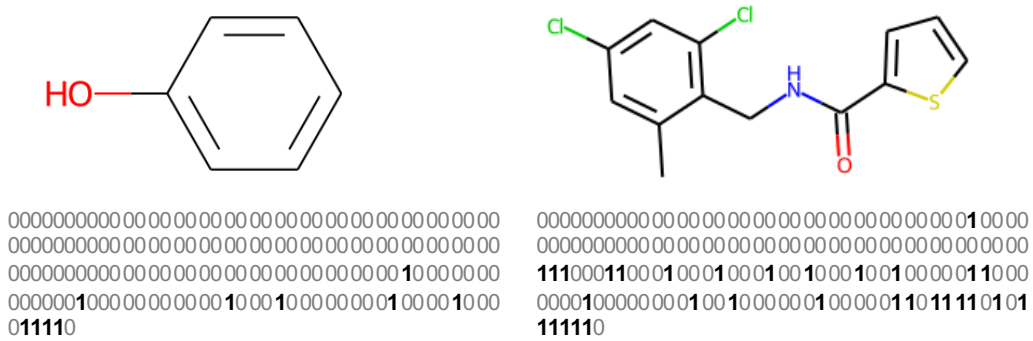


Figure 6: MACCS keys encoding.

Table 6: MACCS keys statistics for pretraining and downstream dataset.

Dataset	Average	S.D	Min	Max
Pretraining dataset	51.99	13.44	1	106
BBBP	46.03	14.48	2	96
SIDER	46.61	17.93	1	105
ClinTox	46.13	16.54	2	92
BACE	61.04	12.53	21	93
Tox21	32.73	16.63	2	99
ToxCast	33.50	17.30	2	101
FreeSolv	15.26	9.78	1	62
ESOL	23.63	15.52	1	76
Lipophilicity	51.61	14.26	7	93
QM7	18.75	7.80	1	50
QM8	23.00	8.10	1	48

Hyperparameter Search Range. For each downstream task, we search for the best hyperparameter combinations. We perform the Bayesian optimization over the validation set and use the hyperparameters for the best validation score to report the test score. The hyperparameter range that we searched over is shown in Table 8.

Pretraining Task. To pretrain our network, we extract the 200 real-valued descriptors for each molecule using RDKit package [38]. This task is proposed by [52]. Through this task, we can make the model to learn the physicochemical properties of the input molecules.

E Parameters and space complexity of different models

We present the number of parameters of transformer-based molecular representation learning models in Table 9. Except for CoMPT, our model has a comparable or lower number of parameters than other transformer-based models.

We also present the space complexity in Table 9. The space complexity for our attention computation is linear to the number of atoms (i.e., $\mathcal{O}(n)$ where n is the number of atoms). On the other hand, the space complexity of other transformer-based molecule representation learning models is quadratic to the number of atoms (i.e., $\mathcal{O}(n^2)$).

Table 7: Model configurations and hyperparameters for pretraining.

	Parameter
M	4
N	3
Model hidden dimension	768
FFN inner-layer dimension	768
# of attention heads	16
Learning rate	0.0001
Epoch	10
Dropout	0.1
Batch size	32

Table 8: Finetuning hyperparameter search range.

Hyperparameter	Description	Range
Learninng rate	The learning rate	0.000001 $\sim$ 0.001
# of MLP layers	The number of last MLP layers	1, 2, 3
Dropout	Dropout ratio	0.0 $\sim$ 0.5
Weight decay	Weight decay	0.0, 0.001, 0.0001, 0.00001

Table 9: Parameter and complexity comparison between different models.

	Parameters	Space Complexity for Attention
GROVER [24]	48M	$\mathcal{O}(n^2)$
Graphormer [8]	47M	$\mathcal{O}(n^2)$
MAT [7]	42M	$\mathcal{O}(n^2)$
CoMPT [27]	2.7M	$\mathcal{O}(n^2)$
Ours	41M	$\mathcal{O}(n)$

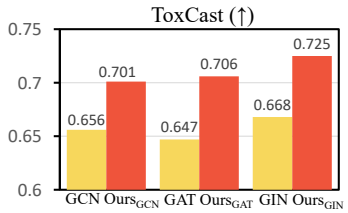


Figure 7: Comparison between standard three GNNs and our model on ToxCast dataset.

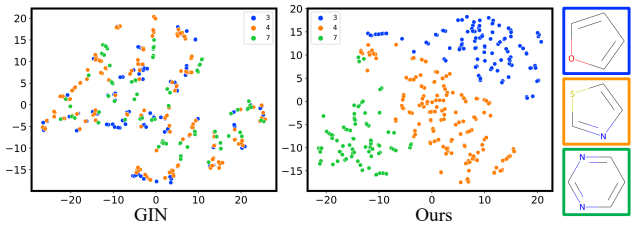


Figure 8: t-SNE embedding of molecules with different substructures.

F Additional Ablation Study

Different GNN branch Our model can flexibly utilize other GNN architectures as our GNN branch. We test the changes in performance when our model design is applied with different GNN architectures. Figure 7 shows the comparison between commonly-used GNNs, i.e., Graph Convolutional Network (GCN), Graph Attention Network (GAT) [53] and GIN, and our models with the GNN branch switched to each corresponding GNN on ToxCast dataset. There is a significant performance improvement when our model is adopted to each GNN model.

Change GNN branch to 3D-aware GNN To observe the effect of the choice of GNN architecture for GNN branch, we conduct additional experiments with 3D-aware GNN. We changed our GNN branch from GIN to SGCN [54] that utilizes 3d coordinates of input molecules. Please note that the two models are not pre-trained and instead trained on the downstream dataset from scratch. Figure 9 shows that the model with 3D-aware GNN shows better performance than our original model on QM7 dataset whose task is closely related to 3D information. The result shows the flexibility of our framework that can be further tuned by using task-appropriate GNN architecture.

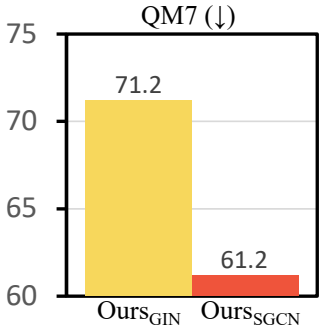


Figure 9: Comparison of our model where the GNN branch is replaced with 3D GNN.

Effectiveness of substructures Figure 8 shows the t-SNE embeddings of molecules with different substructures. Our model discriminates molecules with similar substructures (aromatic rings with one or two different atoms) better than GIN.

Other substructures as input tokens for Transformer Despite our model utilizes MACCS keys for the input of Transformer, our model can flexibly receive any substructure vocabulary. We conduct experiments using other substructures, that are ECFP fingerprints and Tree decomposition. Table 10 shows the results. It shows using MACCS keys achieves better performance than ECFP fingerprint in most of the downstream tasks. We speculate that MACCS keys include predefined functional groups such as the Hydroxyl group (-OH) whereas ECFP encodes local substructure around an atom (i.e., a certain radius neighborhood of an atom). In this aspect, ECFP fingerprint is similar to how GNN encodes node information and as we already utilize GNN to encode local information, ECFP would not bring new information about molecules.

Table 10: Comparison between different substructures applied to our model.

Method	BBBP ↑	BACE ↑	Tox21 ↑	ToxCast ↑	FreeSolv ↓	ESOL ↓	Lipo ↓	QM8 ↓
MACCS key	0.934	0.868	0.818	0.725	2.00	0.878	0.582	0.0140
ECFP 4	0.925	0.869	0.818	0.716	2.52	0.900	0.596	0.0152
ECFP 6	0.903	0.861	0.818	0.709	2.30	0.949	0.592	0.0151
Tree Decomposition	0.925	0.848	0.796	0.715	2.37	0.885	0.614	-

Computation time for graph features Figure 10 shows the time required in millisecond to compute the MACCS keys, shortest path and 3D distance for each molecule with different number of atoms. As shown in the figure, the time required for 3D distance and the shortest path increases dramatically as the number of atoms increase, which make these features impossible to be applied in large molecules. However, identifying the MACCS key substructures does not depend on the input molecule's number of atoms.

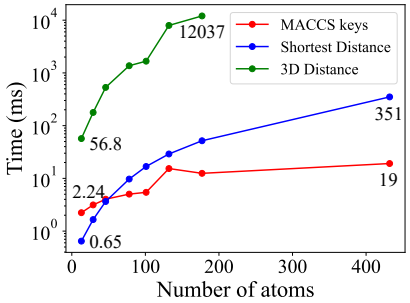


Figure 10: Time required to compute each feature per molecule with different number of atoms.

G Performance on MPG split

As MPG uses different splits to GROVER on the paper, we reproduce the result of our model on the same split that MPG used for fair comparison. As shown in Table 11, our model outperforms MPG on most of the downstream datasets.

Table 11: Comparison between MPG and our model on the split from MPG paper.

Method	BBBP ↑	SIDER ↑	Clintox ↑	BACE ↑	Tox21 ↑	ToxCast ↑	FreeSolv ↓	ESOL ↓	Lipo ↓
MPG	0.922	0.661	0.963	0.920	0.837	0.748	1.269	0.741	0.556
Ours	0.943	0.665	0.958	0.931	0.847	0.748	1.278	0.719	0.546

H Others

URLs for pretrained model for baselines. We use the following sources for pretrained models to reproduce the results.

Graphormer: `https://github.com/microsoft/Graphormer`

MAT: `https://github.com/ardigen/MAT`

GraphLoG: `https://github.com/DeepGraphLearning/GraphLoG`

GEM: `https://github.com/PaddlePaddle/PaddleHelix/tree/dev/apps/pretrained_compound/ChemRL/GEM`